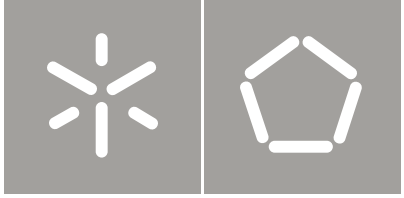
Universidade do Minho
Escola de Engenharia

Marino Fernandes

Interfaces adaptativas para sistemas de gestão de Domótica

Interfaces adaptativas para sistemas de gestão de Domótica
Marino Fernandes
UMinho | 2011

Dezembro de 2011



Universidade do Minho
Escola de Engenharia

Marino Fernandes

Interfaces adaptativas para sistemas de gestão de Domótica

Tese de Mestrado
Ciclo de Estudos Integrados Conducentes ao
Grau de Mestre em Engenharia de Comunicações

Trabalho efectuado sob a orientação do
Professor Doutor Bruno Dias

Dezembro de 2011

Agradecimentos

Gostaria de agradecer ao professor Bruno Dias, pela orientação, dedicação, empenho e paciência durante a realização deste trabalho.

Aos meus pais, por tudo o que fizeram e continuam a fazer por mim.

À minha namorada, Margarida Pereira, pelo apoio, amor, carinho, e paciência demonstrados.

Aos meus colegas e amigos André Silva, Emanuel Freitas e Leandro Duarte, pelas longas horas que passámos juntos e pela contribuição nas diversas discussões que levaram o projecto DOMinho a bom porto.

Ao Miguel Lopes pela ajuda facultada na inicialização da API SNMP4j utilizada neste trabalho.

Por fim, a todos os meus amigos, nomeadamente ao David Silva e ao Hooman Shahrasbi pelos bons conselhos e disponibilidade demonstrada.

Resumo

A utilização da domótica em ambientes domésticos e empresariais tem tido um acentuado crescimento na última década. Os recentes avanços das tecnologias, possibilitam o controlo da maioria dos equipamentos disponíveis em habitações, hospitais, escritórios, etc. Estes sistemas, normalmente muito onerosos, tentam proporcionar aos seus utilizadores uma maior comodidade, segurança, poupança energética, entretenimento, etc. No entanto, é ainda possível potenciar as vantagens destes sistemas, se for considerada a associação de softwares inteligentes aos sistemas actuais de domótica. A introdução de interfaces adaptativos e de processos automatizados inteligentes nestes sistemas permite aumentar a simplicidade e intuição das interfaces de utilizador, ao mesmo tempo que permite a programação de funcionalidades mais completas e complexas.

Este é o propósito do projecto DOMinho, em que a adaptação de interfaces e serviços, bem como a utilização de modos de interacções inteligentes entre o sistema e os utilizadores são as palavras chave. Neste projecto pretende-se promover recentes conceitos em sistemas de domótica, tais como, inteligência, adaptação, modularização, generalidade e normalização.

Em particular, o objectivo desta dissertação passa pela definição e desenvolvimento de dois módulos para o projecto DOMinho, o módulo de utilizadores e o módulo de interfaces. Estes componentes devem suportar os utilizadores e equipamentos interfaces do sistema DOMinho, tendo em conta os referidos princípios que o projecto DOMinho almeja. Estes módulos têm, de ser normalizados e genéricos, de forma a que novas informações possam ser adicionadas ao longo do tempo sem qualquer tipo de remodelação estrutural, permitindo uma retro compatibilidade independente das informações que forem adicionadas.

Toda a normalização protocolar deste projecto tem como base o protocolo de gestão Simple Network Management Protocol (SNMP), utilizando o paradigma das Management Information Base (MIB) como modelo de informação.

Através da implementação e experimentação com um protótipo de uma interface gráfica, integrada na criação de um sistema completo baseado na arquitectura DOMinho, foi possível concluir da validade científica da solução proposta, bem como da eventual adequação à sua utilização em futuros produtos comerciais.

Palavras-chave: Domótica, Domótica Inteligente, Interfaces Inteligentes, SNMP

Abstract

The use of automation at home and in business has witnessed a sharp growth in the last decade. Recent advances in technologies allow monitoring and control for most of the equipments at homes, hospitals, offices, etc. These systems usually are expensive and they attempt to provide users with greater convenience, safety, energy savings and entertainment functionalities. However, it is still possible to maximize the advantages of these systems; so to achieve this goal, a combination of intelligent software and existing home automation systems should be considered. The introduction of adaptive interfaces and intelligent automated processes in these systems can increase the simplicity and the intuitiveness of user interfaces, while the programming features allow us more completeness and complexity.

This is the purpose of the DOMinho project, where the adaptation of interfaces and services, as well as the use of intelligent modes of interactions between the system and users are the keywords. This project aims to promote recent concepts in home automation systems such as intelligence, adaptation, modularity, generality and standardization.

In particular, the main purpose of this thesis is to define and develop two modules for the DOMinho project: users and interfaces. These components must support users and system interfaces equipment taking into account the principles defined by DOMinho architecture. These modules have to be standardized and generalized so new information can be added without any structural remodeling, allowing for backward compatibility independent of the added information. All the normalization protocol of this project is achieved based on the most widely available standard management protocol, the Simple Network Management Protocol (SNMP), using the paradigm of the Management Information Base (MIB) as a model information.

Through the implementation and testing with a complete prototype, including one for a graphical user interface, it is possible to conclude that the proposed solution is valid and also suitable for possible future use in commercial products.

Keywords: Home Automation, Smart Home Automation, Intelligent Interfaces, SNMP

Conteúdo

Lista de Figuras

Lista de Tabelas

Acrónimos

ASN	**A**bstract **S**yntax **N**otation
BER	**B**asic **E**ncoding **R**ules
CLP	**C**ontrolador **L**ógico **P**rogramável
DSSS	**D**irect **S**equence **S**pread **S**pectrum
DS-UWB	**D**irect **S**equence **U**ltra **W**ideband
EDR	**E**nhanced **D**ata **R**ate
HS	**H**igh **S**peed
IEEE	**I**nstitute of **E**lectrical and **E**lectronics **E**ngineers
IR	**I**mpulse **R**adio
UI	**U**ser **I**nterfaces
IUI	**I**ntelligent **U**ser **I**nterfaces
MB-OFDM	**M**ulti-**B**and **O**rthogonal **F**requency **D**ivision **M**ultiplexing
MIB	**M**anagement **I**nformation **B**ase
OID	**O**bject **I**dentifier
PPM	**P**ulse-**P**osition **M**odulation
RFID	**R**adio-**F**requency **I**dentification
SMI	**S**tructure **M**anaged **I**nformation
SNMP	**S**imple **N**etwork **M**anagement **P**rotocol
SGMP	**S**imple **G**ateway **M**anagement **P**rotocol
TDMA	**T**ime **D**ivision **M**ultiple **A**ccess
UDP	**U**ser **D**atagram **P**rotocol
UWB	**U**ltra **W**ide **B**and
WLAN	**W**ireless **L**ocal **A**rea **N**etwork
WPAN	**W**ireless **P**ersonal **A**rea **N**etwork

Capítulo 1

Introdução

1.1 Enquadramento

Para que seja permitida a ligação entre dois elementos, homem e máquina, é necessária uma ponte de ligação. A interface é a responsável por esta ligação no ambiente computacional. Assim, as interfaces são constantemente utilizadas, sendo uma componente importante nos produtos que fazem parte da nossa vida quotidiana.

A evolução do hardware tende a proporcionar um aumento das capacidades funcionais das ferramentas passíveis de utilização em sistemas domóticos, aumentando a complexidade das interfaces disponibilizadas ao utilizador, mas, geralmente, esta complexidade não é desejada uma vez que torna os equipamentos de interface mais caros e difíceis de utilizar. Os sistemas inteligentes e adaptativos permitem combater este aumento de complexidade, simplificando as interacções com o utilizador e ajudando-os a concluir as suas tarefas com maior facilidade [1].

Actualmente já não basta que um produto cumpra as suas funções básicas. É necessário que o seu manuseamento e utilização sejam simples e inequívocos. Essa busca crescente pela qualidade do produto, como forma de melhorar a qualidade de vida do consumidor, tem também estimulado e aprimorado o desenvolvimento de interfaces inteligentes.

Pretende-se assim, no âmbito desta dissertação, criar um sistema de gestão de utilizadores e interfaces que permitam evidenciar os gostos e características pessoais de cada utilizador, de modo a serem tratados de forma individual e única. Para tal, e como os hábitos dos seres humanos estão constantemente a ser alterados, é necessário que o sistema contenha sempre informações

actualizadas acerca dos utilizadores. Por outro lado, as pessoas não se comportam sempre de igual forma em qualquer ambiente, tempo ou local. Neste caso, é pretendido considerar essas alterações comportamentais. O sistema de interfaces, por sua vez, deve normalizar todo o acesso de equipamentos interfaces, disponibilizando um sistema de dados genérico que permita que, equipamentos de diferentes complexibilidades, possam comunicar com o mesmo sistema domótico. Através dum módulo adaptativo torna-se possível a definição dinâmica de menus, tipos de interacção, definição de layouts e estrutura visual nos interfaces. Estas definições dependerão das características e capacidades da interface ligada ao sistema e permitem que os interfaces possam ser adaptados ao utilizador sem que possuam qualquer tipo de software inteligente interno.

DOMinho é a denominação do projecto que pretende integrar estes dois sistemas [2]. Nesta nova arquitectura, visa-se colmatar alguns dos mais relevantes problemas dos sistemas de domótica tradicionais. Para tal, o DOMinho foi desenvolvido de forma a cumprir três requisitos:

- Normalização: obriga a utilização de um tipo especifico de arquitectura e de algumas tecnologias/protocolos normalizados, de forma a que todas as informações possam ser uniformizadas em qualquer sistema que adopte o mesmo modelo.
- Modulação: permite que o sistema seja modular, ou seja, dividido em diferentes camadas permitindo que o desenvolvimento de software possa ser efectuado por diferentes entidades. Torna-se assim possível que módulos desenvolvidos por diferentes fabricantes possam interagir entre si.
- Generalidade: permite que todas as informações do sistema possam ser actualizadas ao longo do tempo sem a necessidade da reestruturação do sistema.

Assim, o modelo preconizado pelo DOMinho é constituído por quatro camadas distintas: dispositivos, adaptação, inferência e serviços, como apresentado na figura 1.1.

- Camada de dispositivos: integra todos os dispositivos externos, nomeadamente os implementados em equipamentos hardware e que já são utilizados nos dispositivos actuais de domótica através de protocolos de comunicação tradicionais, como o X10 e o KNX.

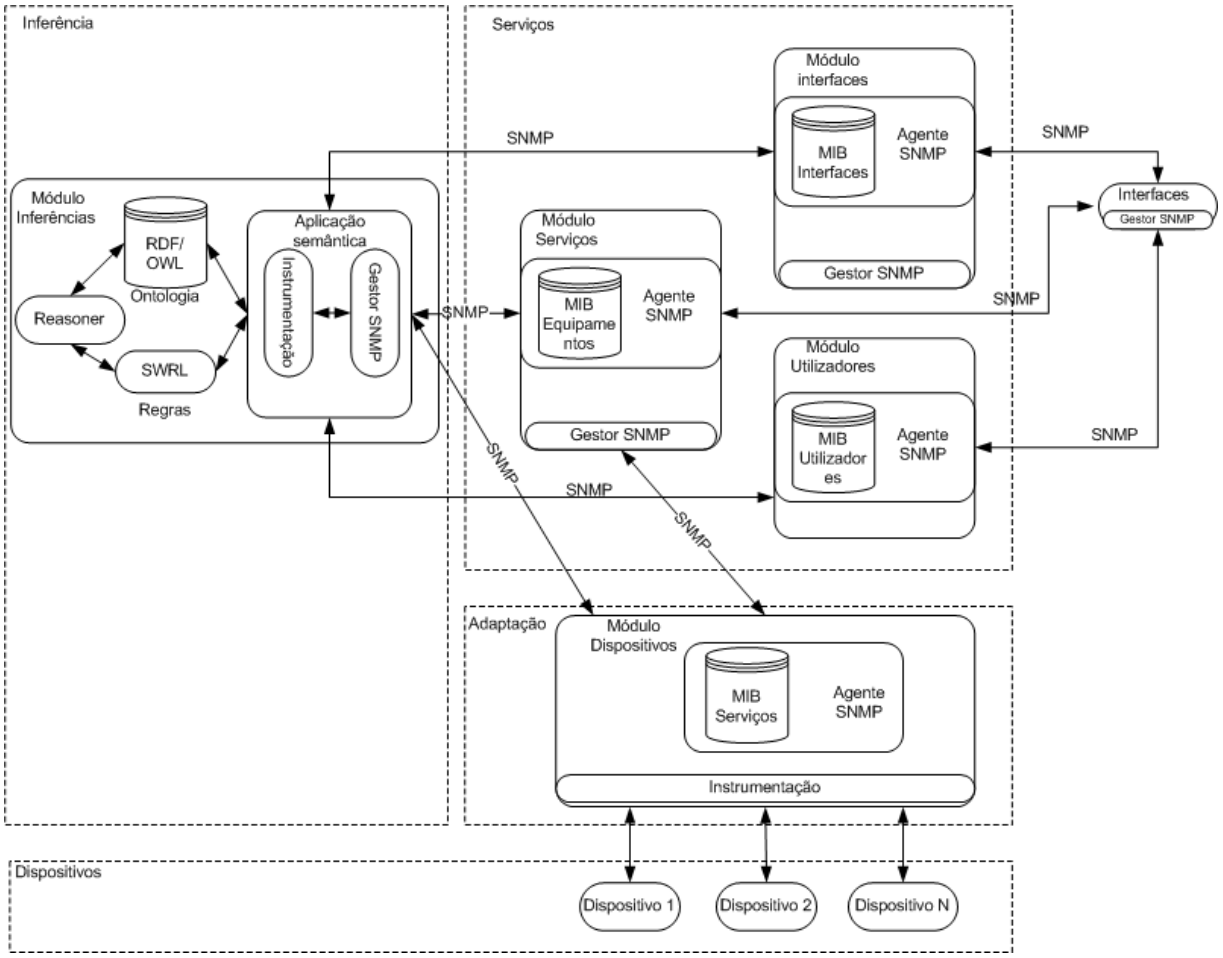


Figura 1.1: Arquitectura do sistema de domótica DOMinho.

- Camada de adaptação: funciona como gateway entre os protocolos usados nos dispositivos de domótica existentes e o protocolo SNMP usado no sistema DOMinho. Esta adaptação permite que todas as camadas possam aceder e controlar as funcionalidades de cada um dos dispositivos através da utilização de um único protocolo de gestão.

- Camada de Serviços: esta camada está dividida em dois componentes principais, os serviços disponíveis (ou serviços produtivos) e o serviço de gestão de utilizadores e interfaces do sistema DOMinho.

  – Serviços produtivos: aqui são geridos todos os equipamentos e dispositivos ligados ao sistema DOMinho. Os serviços disponibilizados por cada equipamento são detectados automaticamente por esta camada tendo em consideração as funcionalidades disponibilizadas. Estes serviços são assim mapeados em actividades, que

são disponibilizadas aos utilizadores através das interfaces. Por outro lado são também aqui geridas todas as localizações dos equipamentos existentes no serviço de domótica.

  - Serviço de gestão de utilizadores e interfaces: fornece um sistema dinâmico de utilizadores, dependente de contextos e que permite conhecer todas as preferências e características dos utilizadores, de modo a que interfaces e serviços possam ser adaptados. Além disso, o sistema de interfaces permite que interfaces de diferentes tipos de complexibilidade possam comunicar com o sistema, de uma forma normalizada e adaptada.

- Camada de inferência: implementa algoritmos inteligentes a aplicar a todo o sistema, suportado, além doutros, nas informações disponibilizadas pelo sistema de utilizadores. Esta camada pode controlar serviços automaticamente e interagir com os utilizadores através da utilização do sistema de interfaces. De frisar que esta é a camada de mais alto nível do sistema, ou seja, esta é a única camada que pode ser retirada, sem que existam problemas de funcionamento do sistema de domótica, ainda que se perdesse uma parte relevante da sua mais-valia.

Toda a comunicação deste sistema é efectuada com recurso ao protocolo de transporte de gestão SNMP. Apesar de normalmente associado à gestão de equipamentos de rede, como routers, é importante perceber que também pode ser utilizado noutros tipos de sistemas. Enquanto que o seu predecessor, o Simple Gateway Management Protocol (SGMP) foi desenvolvido especificamente para internet e encaminhamento IP, o SNMP pode ser utilizado para gerir sistemas Unix, Windows, impressoras, fontes de alimentação, entre outros.

O SNMP constitui uma boa abordagem para a resolução do problema da gestão de redes em ambientes residenciais [3]. Esta conclusão deve-se ao facto dos agentes e do próprio SNMP serem relativamente simples de implementar. Por outro lado, o SNMP utiliza o UDP como mecanismo de transporte, sendo apropriado em ambientes residenciais, onde a fiabilidade é muito alta e não são necessários protocolos orientados à conexão. Por último, o SNMP permite a integração de dispositivos ligados à domótica numa plataforma de gestão de rede complexa e, apesar do seu modelo de informação ser simples, tem um nível funcional suficientemente evoluído para que todo o sistema domótico possa ser implementado. Permite, por exemplo, o envio de notificações automáticas sempre que algo é alterado num campo de uma tabela.

1.2 Objectivos

Esta dissertação visa o desenvolvimento de um sistema de gestão de utilizadores e interfaces que permitam suportar os requisitos de inteligência e adaptabilidade da arquitectura de domótica DOMinho. Este sistema será validado com a criação de um protótipo de uma interface gráfica para domótica que se adapte aos utilizadores, tendo em conta o seu contexto e tendo em consideração as informações dos restantes sistemas que compõem a arquitectura DOMinho. Mais detalhadamente, pretende-se efectuar:

- A análise das principais soluções tecnológicas actuais no mercado para a implementação de terminais de utilizador para gestão de serviços de domótica, visando a identificação e caracterização das suas características funcionais mais comuns e mais relevantes. Este estudo deverá conduzir a um conjunto de conclusões sobre características inteligentes e adaptativas ausentes nos produtos tradicionais e que poderiam ser implementadas tirando partido de tecnologias recentes de comunicação, de localização em tempo real, e da utilização de políticas de interface por contextos.
- Estudo de interfaces inteligentes disponíveis no mercado, bem como a investigação de projectos que utilizem características e soluções de interfaces adaptativos e inteligentes.
- Levantamento de soluções de tecnologias sem fios que proporcionem o bom funcionamento das comunicações entre o comando móvel e o sistema de controlo da casa, bem como uma boa acuidade na localização em tempo real.
- Proposta de uma solução para o sistema de utilizadores e interfaces genérica para dispositivos de gestão de alto nível e sistemas inteligentes, tirando partido de soluções inovadoras no domínio dos interfaces contextualizados (dependentes do espaço, tempo, utilizador e hardware).
- Planeamento, implementação e experimentação com um dispositivo protótipo que permita concluir a validade científica e a utilidade da solução proposta em produtos comercial.

1.3 Estrutura da dissertação

Este documento é composto por 5 capítulos.

No primeiro capítulo é feita uma breve introdução ao tema desta dissertação. É descrita a motivação para a realização deste trabalho, juntamente com os respectivos objectivos a alcançar. Este capítulo termina com a descrição da estrutura adoptada.

O segundo capítulo apresenta o estado da arte relativamente aos sistemas domóticos, interfaces inteligentes, tecnologias sem fios e protocolo SNMP.

No terceiro capítulo são descritos os módulos de utilizador e interfaces propostos, bem como o protótipo que pretende verificar a sua potencial utilização em futuros produtos comerciais de domótica.

No quarto capítulo é apresentada uma descrição do protótipo desenvolvido para o sistema de utilizadores e interfaces, bem como do protótipo da interface gráfica. Neste capítulo é também apresentado um cenário de testes ao sistema DOMinho e os resultados obtidos.

Por fim, no quinto e último capítulo, são apresentadas as conclusões do trabalho desenvolvido.

Capítulo 2

Estado da arte

Neste capítulo será discutido o estado da arte das tecnologias e conceitos mais importantes das áreas associadas aos temas fulcrais desta dissertação. Inicialmente serão analisadas as principais soluções tecnológicas actualmente existentes no mercado de domótica visando as implementações de terminais de utilizador. Pretende-se também analisar qual a postura das empresas, tanto em relação aos métodos mais tradicionais com em relação aos métodos mais inovadores, existentes para a interacção homem máquina neste contexto de aplicação.

Posteriormente serão analisados alguns tipos de interfaces inteligentes por forma a concluir quais poderão ser úteis em ambientes domóticos e qual a aceitação das pessoas quando interagem com as mesmas. Sendo o objectivo desta dissertação o desenvolvimento de um protótipo de um comando móvel para domótica, serão também apresentadas soluções de tecnologias sem fios. A tecnologia mais adequada será escolhida para a implementação no sistema, tendo em consideração parâmetros como a acuidade da localização em tempo real, o consumo de energia e a largura de banda.

2.1 Produtos comerciais

2.1.1 Vivimat: a casa Inteligente

A Vivimat [4] é uma empresa dedicada à área da domótica que desenha, desenvolve e comercializa sistemas de domótica. Esta empresa disponibiliza diferentes tipos de interface para a

gestão do seu serviço de domótica, tais como, teclados, painéis alfanuméricos ou tácteis, TV, PC, PDA, telemóvel, telefone e internet. De todos estes modos de interacção com o utilizador, o equipamento Pantalla Vision Color -7, 2.1 destaca-se, disponibilizando um ecrã táctil, leitor de chaves por proximidade (RFID), activação por presença, leitor de cartões SD, uma interface de infravermelhos que permite a ligação de um comando universal compatível com RC5 para efectuar o controlo do sistema de domótica. O Pantalla Vision Color pode ainda funcionar como vídeo-porteiro, intercomunicador de mãos livres, telefone, moldura de fotos digital e para efectuar acesso à internet.

Figura 2.1: Equipamento interface Pantalla Vision Color -7 [4].

O Wi-Fi e o GSM/GPRS são as tecnologias sem fio utilizadas para interagir com o sistema, através de interfaces proprietários ou através de PDAs e telemóveis. De igual forma, o serviço telefónico deste sistema de domótica pode também ser usado para controlar o sistema de domótica, podendo verificar/controlar alguns sensores/actuadores, ou ser utilizado como um sistema de segurança em caso de um alarme ser activo. Neste caso, até 4 números telefónicos poderão ser contactados e informados através de uma gravação de qual o tipo de alarme accionado.

2.1.2 Control 4

A control4 [5] é uma empresa de domótica que comercializa produtos de domótica residenciais e para a industria hoteleira. Esta empresa possui um vasto leque de interfaces de utilizador, todos

eles de design atractivo, que podem ser vistos na figura 2.2.

Figura 2.2: Equipamento interface Control 4 [5].

Estes interfaces utilizam três tipos de tecnologias de comunicação diferentes, entre elas, duas sem fio, WI-FI e ZigBee, e uma com fio, a tecnologia Ethernet. A tecnologia ZigBee é utilizada no comando remoto com teclado como nos KeyPads. Estes, fornecem botões que podem ser programados pelo utilizador para que este possa rapidamente accionar um determinado serviço, como por exemplo, o controlo de equipamentos multimédia. Esta interface disponibiliza também um serviço de Walkie Talkie.

O tempo de vida das baterias destes interfaces rondam as 6 horas em utilização, e/ou 24 horas em standby. É ainda possível controlar este sistema de domótica através de iPhone, iPad ou iPod touch.

2.1.3 Cardio

A Cardio [6] é uma empresa que comercializa sistemas de domótica, com o objectivo de proporcionar qualidade de vida nas habitações. Esta empresa permite efectuar o controlo da habitação através de consolas fixas, móveis, computadores, PDAs, telemóveis e telefones. Todas as interfaces disponibilizadas permitem ao utilizador uma interacção táctil que, através de uma série de menus e submenus, guiam o utilizador para que, de forma directa, controle a sua casa. Na figura 2.3 pode ser vista uma interface disponibilizada por esta empresa. O controlo baseado em telefones e telemóveis disponibiliza uma voz que guia o utilizador através de um menu de

opções.

Figura 2.3: Equipamento interface Cardio [6].

Dos serviços prestados pela Cardio, é de referir a configuração de luzes para reagir a determinados eventos, como por exemplo, o flash de luzes quando o telefone toca ou o seu controlo automático quando existe a passagem de uma pessoa num determinado espaço da casa. Além da iluminação, este sistema permite também controlar e programar aparelhos existentes na casa, como por exemplo, o funcionamento de uma máquina de lavar roupa para que seja activa a uma determinada hora da noite, visando o aproveitamento da tarifa eléctrica nocturna.

A nível de segurança, a cardio possui um sistema que permite simular a presença de pessoas em casa através da configuração temporal da activação/desactivação de aparelhos, tais como, a televisão e rádio. Quanto ao alarme, assim que despoletado, é efectuada uma chamada telefónica para os números de telefone previamente configurados.

O sistema Cardio oferece, ao utilizador um acesso personalizado que lhe permite efectuar o bloqueio de algumas das configurações disponíveis. Este serviço é disponibilizado perante a autenticação dos utilizadores, por meio de um código de acesso. Por último, é de frisar que esta empresa já comercializou interfaces de interacção por voz, que devido à sua baixa fiabilidade, deixaram de ser comercializados.

2.1.4 Casa do Futuro

A empresa Casa do Futuro [7] desenvolve projectos de telecomunicações e automação residencial e predial. Esta empresa comercializa uma grande variedade de interfaces para a interacção com os utilizadores, entre as quais, interruptores tradicionais, telefones, telemóveis, monitores touchscreen, microcomputador e controlos remotos (Pocket PC, telemóveis, controlos universais, ou acessos através da internet) [8].

Através destas interfaces, é possível a disponibilização de diversos serviços, tais como, um serviço telefónico interno à habitação, um som ambiente integrado com TV/DVD, intercomunicadores, intranet, internet banda larga, vídeo-conferência e telecomando de sistemas. O Wi-Fi, o infravermelho e a rádio frequência são as tecnologias usadas para as comunicações das interfaces remotas. Através da utilização de recursos de automação, as habitações podem interagir remotamente com os moradores, através do envio e recepção de mensagens SMS, e-mails e chamadas telefónicas.

Neste sistema são usados CLPs (Controlador Lógico-Programável) que conferem a "inteligência" às habitações, disponibilizando serviços como o ajuste automático de temperatura de acordo com cada utilizador, o aumento ou diminuição da incidência solar, ou mesmo com outras variações de temperatura externas, gerindo assim o consumo de energia. De igual modo, é possível monitorizar as luzes que estão acesas em toda a casa, programar a iluminação para que possa ser accionada em determinados horários ou simular a presença de pessoas em casa e o accionamento e programação de bombas e filtros (utilizados em piscinas e cisternas).

A nível de segurança, este sistema permite efectuar o controlo da entrada e saída de veículos através da utilização de sensores de proximidade ou de tags para veículos, e permite ainda a utilização de leitores biométricos ou cartões magnéticos para gerir o fluxo de moradores (residentes, visitantes e funcionários) emitindo relatórios, e restringindo o acesso a determinadas pessoas.

Um exemplo prático de utilização deste sistema passa pelo accionamento da fechadura através do reconhecimento da impressão digital. O sistema reage automaticamente, acendendo, por exemplo, uma iluminação especial, colocando a sua música preferida a tocar, accionando o ar condicionado. No caso da ausência do utilizador na habitação, o sistema pode ainda, por exemplo, avisar o morador por e-mail ou mensagem, caso a campainha seja accionada. Assim, o utilizador poderá então ver quem está à porta (via internet) e até mesmo accionar remotamente

a sua abertura.

2.2 Trabalhos relacionados

Neste capitulo, será realizada uma pequena descrição de três trabalhos de domótica não comerciais com características inteligentes.

2.2.1 Amigo

Amigo [9] é um projecto apadrinhado por quinze das principais empresas e instituições de investigação da Europa nas áreas do desenvolvimento de software, redes móveis e domésticas, consumo de electrónica e aplicações domésticas. Este projecto visa o desenvolvimento de um "middleware" aberto, normalizado e com bases de interoperabilidade e de serviços de utilizador atraentes.

O middleware desenvolvido pretende integrar dinamicamente sistemas heterogéneos assegurando a interoperabilidade entre serviços e dispositivos. Por exemplo, electrodomésticos (sistemas de aquecimento, sistemas de iluminação, máquinas de lavar, frigoríficos), equipamentos multimédia e dispositivos pessoais (telemóveis, PDAs) são conectados numa rede doméstica para trabalhar de forma interoperável. Esta interoperabilidade entre diferentes domínios de aplicação pode também ser extendida a habitações e outros locais. Este projecto tem, assim, como um dos principais objectivos fornecer aos utilizadores finais um serviço que lhes permita compartilhar actividades e experiências de uma forma simples e personalizada. Uma visão geral dos componentes do Amigo:

- Programação e desenvolvimento da framework: permite o desenvolvimento de novas aplicações num curto espaço de tempo, retirando por exemplo a necessidade do conhecimento de detalhes específicos de protocolos de comunicação remota.
- Serviço de gestão de contextos: efectua uma aquisição de informação proveniente de várias fontes, tais como sensores físicos e actividades do utilizador, de modo a combinar as informações e fornecer serviços contextualizados.

- Notificação: fornece a funcionalidade básica necessária para o desenvolvimento de aplicações, permitindo a pessoas e outras aplicações saberem de qualquer alteração significativa no contexto com o mínimo esforço. Este componente é capaz de acompanhar as mudanças em vários tipos de contexto, por exemplo, actividades e presença de pessoas.
- Privacidade e Segurança: fornece acesso ao serviço de autenticação e autorização.
- Modelo e perfil de utilizador: fornece a metodologia para melhorar a eficácia dos serviços e interfaces, permitindo a apresentação da informação adaptada ao utilizador e ao contexto e infere o seu comportamento futuro. Deste modo encontra informações relevantes, e adapta os recursos da interface ao utilizador e ao contexto em que é usada.

Toda a adaptação é possível devido à construção, manutenção e exploração de modelos e perfis de utilizador, tornando-se representações explícitas das preferências individuais dos utilizadores. Neste sistema, o utilizador pode interagir com o sistema através da utilização de um gestor de diálogo multimodal, ou através de interacção de voz ou gestual.

2.2.2 Ambient intelligence application in home automation

A empresa Fagor tem vindo a trabalhar num sistema de domótica que comunica através de uma rede power-line [10].

Todos os equipamentos da habitação são ligados a um controlador central chamado Maior-Domo que gere todo o sistema. Esta aplicação permite que o utilizador dialogue e requisite funcionalidades ao sistema de domótica como se de um amigo se trate. Quando o utilizador fala, o Maior-Domo verifica a existência de comandos nas ordens vocais dadas pelo utilizador, e em caso afirmativo, o sistema extrai os diferentes comandos e controla os dispositivos da casa.

Este sistema torna-se, assim, muito flexível e intuitivo para o utilizador, dado que este não necessita de dar comandos específicos para realizar determinada acção. Através da linguagem natural e sem qualquer necessidade de aprendizagem o utilizador pode controlar toda a habitação. Da mesma forma, qualquer tipo de informação que advenha de um dispositivo para o utilizador, pode ser transmitido através de voz. Assim, todos os dados captados pelos microfones são filtrados, de forma a que os ruídos sejam atenuados. Depois de filtrados, o sistema de reconhecimento de voz processa a frase proferida pelo utilizador. Um número bastante alargado de

frases, chamado de "grammar", compõe o diálogo possível entre uma pessoa e todo o sistema. Deste modo, o utilizador pode interagir com o sistema de diferentes formas, tais como, com a utilização de uma série de expressões, falando natural e espontaneamente ou mesmo dialogando com o sistema.

2.2.3 Location-aware services and interfaces in smart homes using multi-agent systems

Neste trabalho [11] é proposta uma arquitectura para a construção de uma casa inteligente utilizando sistemas multi-agentes. Estes sistemas providenciam uma framework distribuída e flexível que comunica e negoceia mecanismos entre os seus componentes de forma a permitir-lhes tomar decisões complexas de forma eficaz e eficiente.

É assim apresentada uma arquitectura que permite que um único controlo remoto universal seja capaz de gerir todos os equipamentos do sistema. Este comando apresenta interfaces dependentes dos serviços existentes no local onde se encontra e permite que o conteúdo multimédia siga o utilizador caso este troque de divisão da casa. Através da utilização do serviço de localização, são ainda registadas as preferências dos utilizadores com dependência da sua localização.

O sistema de detecção e localização do utilizador é baseado na tecnologia Bluetooth. Esta tecnologia foi escolhida devido à sua ubiquidade, podendo ser encontrada numa grande variedade dispositivos móveis, tais como telemóveis e PDAs. No entanto, foi concluído que o processo de localização revelou-se mais lento, menos fiável, e menos eficiente em termos de consumo de energia, do que os autores esperariam.

2.3 Interfaces de utilizador

O conceito de interface é amplo, podendo-se expressar como um meio que permite a interacção entre um utilizador e uma máquina, sendo através deste que o utilizador consegue interagir com os sistemas existentes. Deste modo, a interface é o primeiro contacto que um utilizador tem com um sistema informático. Este impacto pode influenciar desde logo a opinião do utilizador, ou seja, o sistema pode ter grandes capacidades e muitas funcionalidades, mas se a sua interface não se reflectir simples, atractiva e flexível, pode ser criada uma imagem negativa desse mesmo

sistema. Do mesmo modo, o sistema pode ser fraco mas, caso tenha uma interface atractiva favorece um primeiro impacto positivo com o utilizador. A criação de interfaces diferenciadoras é um desafio, por ser difícil ou mesmo impossível agradar a todos os possíveis utilizadores.

A evolução tecnológica, a necessidade do aumento de segurança e a necessidade de uma maior facilidade de aprendizagem, têm dado origem a maiores esforços no estudo, investigação e desenvolvimento no campo da interacção "homem-máquina". Grande parte da população mundial interage diariamente com sistemas informáticos. Este aumento do número de pessoas a interagir com computadores também aumenta as diferenças relativas aos níveis de conhecimento e experiência que possuem com estes sistemas, sendo necessária a criação de interfaces centradas no utilizador e não apenas na aplicação. É assim necessário o desenvolvimento de interfaces cada vez mais intuitivas, personalizadas e naturais.

Em [12] são referidos uma grande variedade de factores com os quais os designers de interfaces de utilizadores se devem preocupar. Nestes factores são referidas ideias como, o que as pessoas querem e esperam do sistema, quais são as suas limitações físicas e habilidades, qual a sua percepção do sistema de processamento de informações e o que as pessoas acham mais agradável e atraente. Estes factores levam, desde logo, a concluir que desenvolver uma interface de utilizador é complicado e que se devem ter em atenção vários factores decisivos para a sua aceitação. Conclui-se assim, que desenvolver um sistema que permita diversas formas de interacção aumenta a satisfação de quem o utiliza, pois permite a escolha da interacção que mais convier ao utilizador.

2.3.1 Interfaces inteligentes

As interfaces de utilizador inteligentes são um sub-campo da interacção homem-máquina. Estas interfaces visam aumentar a flexibilidade e o poder de interacção entre humanos e sistemas informáticos [13], através da resolução de alguns dos problemas de manipulação directa que as interfaces tradicionais não conseguem resolver. As interfaces inteligentes fornecem as seguintes vantagens [14, 15]:

- Criação de sistemas personalizados: as interfaces de utilizador inteligentes (IUIs) baseiam-se nas preferências, hábitos e comportamentos dos utilizadores para fornecer métodos de interacção personalizados.

- Filtragem do excesso de informação: uma IUI pode filtrar informações irrelevantes para os utilizadores, reduzindo a quantidade de informações disponibilizadas e reduzindo a sua carga cognitiva.

- Ajuda em novos programas: as IUI podem ajudar os utilizadores a interagir com os sistemas, fornecendo informações que permitam simplificar as tarefas, explicando novos conceitos e corrigindo más utilizações.

- Reconhecer tarefas do utilizador: uma IUI pode entender e reconhecer as acções dos utilizadores, reagindo automaticamente às mesmas, permitindo que os utilizadores se concentrem noutras tarefas.

- Diversas formas de interacção: as IUI fornecem novas técnicas de interacção, como por exemplo, a utilização da voz e de gestos para interagir com os sistemas. Estas novas interacções simplificam a utilização dos sistemas e possibilitam que pessoas com deficiência possam utilizar sistemas informáticos com maior complexibilidade.

Assim, de modo a que estas vantagens possam ser concretizáveis, existe a necessidade dos interfaces conhecerem os utilizadores. É por isso essencial recorrer a técnicas que ajudam a conhecer e compreender os utilizadores [15]:

- Tecnologias Inteligentes: técnicas inovadoras para obter informação dos utilizadores, como por exemplo, linguagem natural (reconhecimento de voz e sistemas de diálogo), reconhecimento de gestos, etc;

- Modelos do Utilizador: técnicas que permitem manter ou inferir conhecimento sobre um utilizador baseando-se nos dados recebidos;

- Adaptabilidade ao Utilizador: inclui técnicas que permitem que a interacção homem-máquina seja adaptada a diferentes utilizadores e contextos de utilização;

- Criação de Esclarecimentos: técnicas que permitem que um sistema explique aos utilizadores os seus resultados.

- Personalização: para personalizar as interfaces, as IUIs recorrem normalmente a modelos de utilizador, onde são encontradas informações sobre os conhecimentos e competências

dos utilizadores. Novos conhecimentos podem ser inferidos a partir das entradas e do histórico da interacção do utilizador com o sistema.

- Flexibilidade de utilização: as IUIs utilizam normalmente sistemas dinâmicos para adaptação e aprendizagem de informações. Assim, é possível a adaptação através de inferências ao conhecimento armazenado no modelo do utilizador e a aprendizagem de novos conhecimentos através de inferências sobre os novos dados recebidos.

Através da utilização das IUIs, é possível alterar o conceito das interfaces tradicionais, tornando-as como uma espécie de ser humano que se encontra do outro lado do sistema, que pode interagir com o utilizador através de linguagem natural e que conhece as suas preferências. São, assim, facilmente distinguíveis das interfaces não inteligentes, pois as últimas não reflectem as diferenças inerentes aos seus utilizadores. Por outro lado, apresentam menus fixos, oferecendo, muitas vezes tipos de interacção já obsoletos, funcionando apenas como meio de comunicação entre um ser humano e uma máquina ou sistema informático.

2.3.2 Interfaces adaptativas

A adaptabilidade é uma das principiais funcionalidades exigidas às interfaces inteligentes de utilizador. Este tipo de interface tem-se apresentado promissor na tentativa de superar os problemas actuais de complexidade na interacção homem-máquina. A interface adaptativa é definida [16] como "um pedaço de software que melhora a capacidade de interagir com o utilizador através da construção de um modelo baseado na experiência parcial desse utilizador". Esta definição mostra claramente que uma interface adaptativa é desenhada com base no comportamento do utilizador. Estas interfaces são ainda caracterizadas pela adaptação personalizada a um utilizador específico, uma nova forma de interface que auxilia o utilizador a personalizar os seus conteúdos [1]. A existência de sistemas de informação cada vez mais complexos, o grande crescimento do número de dispositivos e das suas funcionalidades e as diferentes características dos utilizadores (interesses, habilidades, experiência e conhecimento) torna ainda mais importante a adaptabilidade de interfaces. Com a sua utilização é possível minimizar a sobrecarga de informações, filtrar informações irrelevantes, antecipar as necessidades e tomar a iniciativa de fazer sugestões aos utilizadores, recuperar informações relevantes, adaptar as interfaces às necessidades de um determinado utilizador, aprender os seus comportamentos (novos conceitos, técnicas), fornecer

outras formas de interacção com os sistemas, como por exemplo a utilização da voz ou de gestos, entre outras [13, 16–18].

Grande parte das interfaces dos sistemas actuais não são baseadas no utilizador, mas sim no sistema [19]. Dificulta-se a interacção dos utilizadores se forem ignoradas as diferenças dos níveis de experiência dos utilizadores, os estilos de aprendizagem, as habilidades cognitivas, os seus antecedentes, a sua personalidade, a sua motivação, as suas preferências e os ambientes de utilização [20]. Uma forma de melhorar a interacção é adaptar as interfaces à experiência média de todos os utilizadores. Outra abordagem possível é fornecer ajuda e assistência aos utilizadores, de forma a estes serem aconselhados, tendo em consideração as funcionalidades requeridas e onde são requeridas. No entanto, ambas as abordagens têm desvantagens, a primeira assume uma interface que se adequa a todos os utilizadores (o que é pouco provável, pois os utilizadores possuem preferências distintas) a segunda abordagem proporciona o mesmo conselho a diferentes utilizadores, ou seja, ignora os diferentes níveis de experiência que estes possuem. A criação de interfaces adaptativas dependentes das preferências de cada utilizador solucionam estes problemas. Estas interfaces são baseadas nas experiências individuais das interacções actuais e passadas do utilizador. Assim, uma interface adaptativa pode ser descrita como "um conjunto de ecrãs e comandos, um operador humano, um sistema de software capaz de modificar a apresentação das informações, e a atribuição de tarefas a serem realizadas em função do estado do operador, do estado do sistema, e do ambiente em que tanto o operador e o sistema estão inseridos" [20].

Existem duas técnicas muito utilizadas para adaptação [1]:

- Apresentação adaptativa: personaliza o conteúdo apresentado ao utilizador, tendo em consideração o nível de conhecimento do utilizador sobre determinado assunto;
- Navegação adaptativa: personaliza os caminhos que os utilizadores têm de seguir para concluírem as suas tarefas na aplicação, escondendo determinadas funcionalidades do sistema consoante a experiência do utilizador.

Estas técnicas fornecem vantagens distintas aos utilizadores. A criação de interfaces onde a conjugação destas técnicas seja realizada resulta numa melhoria evidente em relação às interfaces não inteligentes. É necessário que os sistemas se adaptem às necessidades, habilidades,

capacidades, preferências e contexto dos utilizadores, assim como às suas expectativas e objectivos [21].

2.3.3 Adaptação ao utilizador

Para que um sistema se possa adaptar a um determinado utilizador, é necessário que este o conheça. Torna-se assim imprescindível um mecanismo que permita extrair informações dos utilizadores. Existem pelo menos duas formas de o conseguir: questionar o utilizador sobre as suas preferências ou inferir automaticamente os seus hábitos. O primeiro método é simples, e pode ser realizado através da utilização de questionários ou templates disponibilizados para o efeito. No entanto, este tipo de solução é incómoda para o utilizador e pode nem sempre ser a mais correcta, pois o sistema não sabe se o utilizador responde correctamente aos questionários propostos. O segundo método evita os problemas do anterior e melhora a personalização do processo a fim de torná-lo mais fiável. Este processo é feito através da monitorização das acções dos utilizadores de forma a que posteriormente possam ser compreendidas e adaptados os serviços. Este último processo possui, no entanto, a desvantagem de necessitar de um longo período de tempo para recolher informações suficientes sobre o utilizador de modo a que possa efectuar adaptações com um bom grau de confiança. No entanto, esta desvantagem pode ser atenuada através da utilização de estereótipos.

Estereótipos são, um conjunto de preferências que podem descrever um certo grupo de utilizadores, podendo ser muito gerais ou muito específicos [22].

Através da utilização de estereótipos, é possível atribuir um conhecimento médio de cada um dos utilizadores ao sistema, para que este os possa utilizar enquanto não têm um conhecimento específico desses utilizadores. Apesar das vantagens e desvantagens de cada um dos métodos referidos, podem ser encontradas algumas técnicas para tornar mais fácil e prática a personalização da interface. Caso as acções do utilizador não sejam significativas para se poder proceder a uma boa adaptação da interface, esta poderia sugerir um conjunto de alternativas de adaptação ao utilizador, podendo este então optar por uma delas. Assim, não sendo este método de conhecimento automático nem totalmente manual, pode ser considerado como um mecanismo de "meio-termo" e pode utilizar os seguintes processos [23]:

- Auto-adaptação: a interface realiza todas as adaptações sem interferência do utilizador.

- Utilizador inicia a auto-adaptação: o utilizador inicia o processo de adaptação, sendo a interface responsável pela proposta de alternativas, a selecção da melhor opção e execução de todo o processo.
- Utilizador controla a auto-adaptação: a interface é responsável pela iniciativa e proposta de alternativas, sendo o utilizador a decidir qual a opção pretendida.
- Sistema auxilia adaptação: o utilizador inicia a adaptação, sendo deixado à interface a responsabilidade da apresentação de algumas alternativas. Após a escolha do utilizador, a interface adapta e executa a opção pretendida.
- Sistema inicia adaptação: a interface inicia o processo de adaptação, sendo deixado ao utilizador a responsabilidade de propor alternativas e seleccionar a opção pretendida. Neste caso, a execução da adaptação pode ser realizada pelo utilizador ou pela interface.
- Adaptação: Todo o processo de adaptação é deixado ao cargo do utilizador, excepto a sua execução, que é realizada pela interface.

2.3.4 Perfil de utilizador

O perfil de utilizador representa as características de um utilizador ou grupo de utilizadores, fornecendo ao sistema um conhecimento básico das suas preferências [24]. Serve como elemento central à maioria dos sistemas adaptativos e personalizados [25]. Por outras palavras, o perfil de utilizador por ser definido como um meio utilizado para melhorar a capacidade de interacção com o utilizador através da utilização do conhecimento parcial que este possui do utilizador, tais como, idade, nível de conhecimento de informática, educação, idiomas falados, entre outros. Em geral são necessários três passos para incluir os perfis de utilizador num software [24]:

- armazenar os dados referentes ao utilizador;
- inicializar a leitura desses dados e configurar variáveis adequadas no sistema;
- seleccionar alternativas para o utilizador, baseando-se nos valores das variáveis criadas.

2.3.5 Modelos de utilizador

Os modelos de utilizador são importantes ferramentas de personalização da sub-área da interacção homem-computador, onde são representadas características de utilizadores (como personalidade, habilidades, necessidades, preferências, etc). Este modelo permite a representação dinâmica de um determinado utilizador, preocupando-se com a aquisição, representação e evolução do conhecimento de um dado utilizador [24, 26, 27]. Por vezes, modelo de utilizador e perfil de utilizador são conceitos erroneamente utilizados de forma indiscriminada [28]. O perfil do utilizador contém as informações pessoais do utilizador sem que estas sejam inferidas ou interpretadas, enquanto que o modelo do utilizador expressa a visão geral e abstracta do aprendiz, sendo esta técnica mais sofisticada, permitindo a alteração dinâmica do comportamento do sistema em resposta às actividades do utilizador durante a sua interacção [24].

Os modelos de utilizadores podem ser estáticos ou dinâmicos. Os modelos estáticos são criados à priori pelo designer do sistema e são baseados num "utilizador médio" criado a partir da análise de preferências, crenças, saúde, religião, entre outros conhecimentos. Este conhecimento pode ser fornecido através da utilização de estereótipos. Já um modelo de utilizador dinâmico necessita de estar sempre a ser actualizado, recorrendo para tal, a dados referentes às interacções dos utilizadores.

2.3.6 Tipos de interacções inteligentes de utilizador

As interfaces de utilizador têm sofrido uma grande evolução nos últimos anos. Actualmente existem diversas formas de interagir com uma máquina, cada vez mais simples, naturais e intuitivas.

Uma interface intuitiva entre homem e máquina é aquela que requer pouco treino e profere um estilo de trabalho como aquele usado pelo ser humano a interagir com os ambientes e objectos do seu dia-a-dia. O homem procura, explora, manipula, fala, ouve e movimenta-se utilizando as suas habilidades naturais como lhe são apropriadas e aplicadas a cada tarefa [29]. Estas habilidades também podem ser utilizadas para interagir com sistemas computacionais utilizando interfaces inteligentes.

2.3.6.1 Interfaces hápticas

As interfaces hápticas representam a busca de dispositivos que possam traduzir as sensações do mundo virtual para o mundo físico, através da geração de sinais mecânicos que estimulam a interacção dos humanos. Estas sensações são conseguidas através de dispositivos hápticos que fornecem sensações de contacto com ambientes gerados por computador de modo a que, quando os objectos virtuais sejam tocados, eles pareçam reais. Estes dispositivos permitem aumentar a sensação de realismo fornecendo resposta táctil, restrição de movimentos ou aplicação de forças em vários graus de liberdade [30]. Uma interface háptica é a combinação de dois tipos de sensações, "*cutaneous*" e "*kinesthetic*" (incluindo "*proprioception*") [29, 31, 32]. As sensações "*cutaneous*" exploram percepções tácteis, que são normalmente transmitidas através da pele (sensações de temperatura, dor, pressão e textura), habitualmente usadas para indicar se um utilizador se encontra ou não em contacto com um objecto virtual ou para simular a textura de um objecto. Por sua vez, as sensações "*kinesthetic*" surgem nos músculos e tendões, dando a sensação de uma força a ser aplicada. Assim, interagem principalmente sobre a forma de manipuladores robóticos, funcionando como mais um canal independente para o cérebro, cuja sua informação é assimilada inconscientemente [29]. Este canal provoca uma maior quantidade de informação a processar pelo cérebro, o que reduz o erro e o tempo necessário para completar uma tarefa. Já as sensações "*proprioception*" permitem a percepção da posição dos membros e do movimento do corpo.

Actualmente, através da utilização das sensações "*cutaneous*" pode ser oferecido um grande auxilio a pessoas cegas ou com deficiência visual, existindo várias interfaces hápticas deste tipo para acessibilidade. A interface "Tactuador" fornece essa capacidade aos utilizadores, permitindo que duas pessoas com falta de visão possam conversar através do reconhecimento de diferentes padrões conseguidos através da variação da taxa e amplitude da vibração [29]. Segundo a mesma fonte, é também possível permitir o diálogo entre pessoas com deficiência auditiva e cegos surdos através da reprodução de sons fónicos com diferentes padrões de vibração. Outro exemplo da utilização deste tipo de interface, é o telemóvel da Samsung, modelo Anycall Haptic, pois apesar de ter um visor touch-screen, dá a sensação que se está a interagir com um teclado normal [33, 34]. Existem também coletes para jogos que utilizando esta técnica permitindo que o utilizador possa sentir todos os golpes do seu personagem [35].

2.3.6.2 Interfaces tácteis

As interfaces tácteis são utilizadas para comunicar informações através do tacto. Permitem uma interacção mais realista, intuitiva e fácil com os sistemas [36], transmitindo sensações como, calor, pressão e dor, e feedbacks como vibração, frequência, ritmo, força e textura, e braile [37].

Existe um grande leque de aplicações que pode beneficiar deste tipo de interface, como por exemplo, o treino virtual para cirurgiões, toque em materiais remotamente via internet, indústria automóvel, percepção de textura, visualização de dados científicos, ajudas à navegação, etc [38]. Através do uso deste tipo de interfaces é possível melhorar a performance do utilizador [36], onde a sua utilização num telemóvel melhorou 22% da performance dos utilizadores. Foi ainda comprovado que metade dos utilizadores preferem feedback táctil, por causa da melhoria da experiência de utilizador. A falta deste tipo de feedback pode dificultar a utilização eficaz dos dispositivos touch-screen, especialmente quando os utilizadores são incapazes de ver os seus monitores devido à sua dimensão. Em [37] foram realizadas duas experiências demonstram que os utilizadores podem distinguir dez padrões de vibração, com uma percentagem de precisão a rondar os 90%. Neste estudo, concluiu-se ainda que pessoas com deficiência visual distinguem os padrões de vibração utilizados, com tanto sucesso como os participantes sem deficiência visual. Os autores de [39] mostram que o feedback táctil pode reduzir erros e aumentar a velocidade quando se digita num ecrã touch-screen. Interfaces tácteis podem ser usadas para transmitir avisos e erros de serviços utilizando diferentes feedbacks consoante determinado evento. Estes feedbacks permitem ao utilizador sentir e reconhecer informações sem a necessidade de interromper a sua actividade [36].

2.3.6.3 Interfaces de voz

A interface de voz permite uma interacção natural, fácil, prática e intuitiva entre homem e máquina. Estas interfaces são uma forma prática de interagir em pequenas tarefas, onde a cozinha é salientada como a divisão onde se podem tirar maiores vantagens [40]. Já em [41], esta interface é referida como sendo rápida e prática, com poucos erros, uma vez que são poucos os passos a realizar para se chegar ao objectivo final. Além destas vantagens, estas interfaces possibilitam o controlo de dispositivos sem a necessidade do utilizador se deslocar e de existir contacto visual e/ou físico com a interface. Esta característica torna-se mais importante para

pessoas com deficiência visual, ou motora. Num estudo realizado para o controlo de iluminação numa habitação [42] foram comparados diversos tipos de interfaces, como por exemplo, interacção através de texto simples, interacção gráfica, de voz, de voz com localização e de voz com localização gestual. Foi concluído que a utilização da voz é esperada em ambiente inteligentes e que este tipo de interacção tem diferentes níveis de aceitação dependendo da divisão da habitação (contexto de utilização).

Noutro estudo [41] foi comparada a preferência entre uma interface de voz e de uma interface de navegação para controlo de um sistema multimédia. Foi identificado que os utilizadores preferem a interface de voz, mas, no entanto, foi concluído que esta interface deve funcionar como um complemento à interface normal, possibilitando ao utilizador mais do que um tipo de interacção. Além de todas as vantagens concebidas pelas interfaces de voz, existem também desvantagens que lhes são associadas.

A existência de muito barulho no espaço onde o comando é dado pode fazer com que este não seja executado ou seja mal interpretado pelo interface. Por exemplo, este problema é identificado [43] onde a voz do utilizador pode ser atenuada pelo som de um filme ou de uma música. Este tipo de problema é abordado em [44], onde é apresentado um protótipo que demonstra valores de precisão na ordem dos 90%. Por outro lado, existem contextos em que este tipo de interfaces não são de todo aconselhados, como por exemplo em locais como hospitais, bibliotecas, igrejas, museus, entre outros, onde o silencio é a palavra de ordem. Por último este tipo de interface também pode ser útil como interface de saída. Neste sentido, o sistema pode questionar, avisar ou dar alguma informação ao utilizador através deste tipo intuitivo de interacção. Neste caso, este tipo de interacção seria também uma mais-valia para pessoas com deficiências, motoras ou físicas.

2.3.6.4 Interfaces gestuais

A interface gestual, tal como a interface de voz, permite uma interacção simples, natural e intuitiva. Esta interface consiste na monitorização do movimento do corpo humano e na interpretação do comando que dado movimento significa [45]. Embora esta interacção ofereça uma forma menos precisa na interacção do que os métodos tradicionais, a maioria dos sistemas prevêem uma precisão na ordem dos 90% na detecção de gestos [46]. A sua utilização permite referências espaciais e descrições concisas, possui também vantagens sobre a interface de voz,

em meios ruidosos, onde o utilizador pode realizar determinada acção através de simples comandos com a sua mão [43, 47]. A interface gestual possibilita ainda a análise dos movimentos dos utilizadores, podendo fornecer dados diários a médicos, permitindo a prevenção e detecção de problemas graves.

Existe também a possibilidade de utilização de interfaces de voz combinadas com gestos. Num estudo realizado a 19 participantes para resposta a "caixas de diálogo" [48], conclui-se que em 60,4% do tempo foram utilizados gestos da cabeça, enquanto que 20.9% usaram o rato e 18.6% o teclado. Estes tipos de gestos podem assim ser também utilizados para os utilizadores responderem a simples questões de sim ou não realizados por sistemas informáticos. Este tipo de interface pode também ser combinado com outros métodos de interacção ou sistemas [43]:

- Gesto combinado com voz: torna possível a utilização de um único gesto para diferentes funções, pois o utilizador terá de dizer o nome do dispositivo que quer controlar e só depois realiza o gesto para indicar que tipo de comando deseja dar a esse dispositivo.
- Gesto combinado com orientação: o utilizador terá de indicar através de um gesto indicativo/apontador, qual o dispositivo alvo, e depois poderá realizar o gesto para o controlar. Deste modo poderá ter o mesmo tipo de gesto para diferentes controlos em diferentes dispositivos.
- Gestos combinados com localização: consoante a sua localização, o utilizador, pode ser obrigado a deslocar-se para o local onde se encontra o dispositivo alvo para o poder controlar. Neste caso, pode ser utilizado o mesmo gesto para controlar diferentes dispositivos.
- Gestos combinados com detecção direccional: neste caso o utilizador pode controlar o dispositivo para o qual está direccionado.

2.3.6.5 Interfaces multimodais

Uma interface multimodal é caracterizada por permitir múltiplos canais de entrada na interacção homem-computador. Estas interfaces tentam integrar fala, linguagem corporal, gestos, movimentos dos olhos ou dos lábios e outras formas de comunicação, a fim de compreender melhor o utilizador e comunicar de forma mais eficaz [49]. A sua utilização permite o aumento da usabilidade e acessibilidade das interfaces, fazendo com que cada utilizador possa interagir

com os sistemas através de diversos meios de comunicação. Assim, torna-se mais fácil agradar a um maior número de utilizadores devido aos diferentes tipos de interacção disponibilizados. Num estudo com pessoas de idades entre os 50 e os 80 anos de idade, onde é possível controlar os serviços disponibilizados através de uma interface multimodal [50], 95% dos entrevistados consideraram o sistema fácil de utilizar devido à sua interface. As pessoas inquiridas foram ainda questionadas sobre o que fariam caso o sistema não as entendesse. Nesta situação, apenas 5% dos entrevistados responderam que não utilizariam mais o sistema em caso de erro. Os restantes entrevistados admitiram continuar a tentar até que o sistema funcionasse. Esta percentagem mostra que devido às vantagens fornecidas pelos novos tipos de interacção inteligentes, os utilizadores estão disponíveis para as utilizar, mesmo que estas possam conter mais erros do que as tradicionais.

2.3.6.6 Interfaces tangíveis

Uma interface tangível possibilita o uso de objectos físicos (chamados de objectos tangíveis) na interacção com computadores, permitindo uma representação física e digital simultânea [51]. Ou seja, trata-se do uso de objectos físicos para interagir com sistemas de computadores, oferecendo interfaces de função específica e envolvendo habilidades mais humanas do que apenas as cognitivas [52].

Estas interfaces dão forma física à informação digital, fazendo com que quando é movido um objecto numa interface tangível, é automaticamente movido esse objecto físico e a sua representação digital ao mesmo tempo. Um exemplo destas interfaces pode ser analisada em [53], no qual, a partir de um cubo e de movimentos físicos num espaço 3D se pode mudar os canais de televisão. Deste modo, o movimento que é realizado pelo utilizador no cubo é visualizado numa televisão através de um cubo virtual. Nesse cubo é possível visualizar o que está a ser transmitido em até três canais de televisão ao mesmo tempo. Se o cubo for pousado o canal de televisão seleccionado preencherá todo o monitor.

2.3.6.7 Interfaces touch

Interfaces touch permitem a interacção com os sistema através de toques num simples monitor. Através da sua utilização, a existência de um rato ou de um teclado torna-se desnecessária, pois

tudo pode ser controlado através do toque. Os "displays" podem, assim, ser maiores, pois não existe a necessidade de botões. Os resultados de duas experiências [54] demonstram que os utilizadores beneficiam do toque em relação ao uso do rato em "displays" colocados em mesas. No entanto, estes estudos também indicaram que o rato pode ser mais adequado para a interacção de um único utilizador com os displays, pois requerem um único ponto de interacção. Estas interfaces trazem vários benefícios sobre as interfaces tradicionais por se revelarem mais naturais do que trabalhar indirectamente com um rato ou outro dispositivo apontador. No entanto, os ecrãs touch-screen podem apresentar botões pequenos, o que pode dificultar a interacção a quem tem dedos grandes ou dificuldades de visão. No entanto, através da utilização de um feedback táctil esta situação pode ser atenuada [55]. Touch-screens em dispositivos móveis são vantajosos, na medida em que cada entrada das aplicações pode ser personalizada, e a utilização do espaço do visor é mais flexível. Além disso, o display pode ser muito maior, uma vez que teclado físico não é necessário.

2.4 Tecnologias sem fios

As tecnologias sem fio constituem-se como uma alternativa às redes convencionais com fio, fornecendo as mesmas funcionalidades mas de forma flexível e de fácil configuração. Apesar de todas as vantagens inerentes a este tipo de redes, existem contudo desvantagens, como o consumo de energia, o alcance, as interferências no sinal, a menor largura de banda, entre outros.

2.4.1 Bluetooth (IEEE 802.15.1)

O Bluetooth [56–58] é uma tecnologia de área pessoal sem fios (WPAN) de baixo custo, que permite comunicações de curto alcance, geralmente 10 metros com baixos consumos de energia. Esta tecnologia possui 3 versões. A referir, a versão 2.1 + EDR (Enhanced Data Rate), que possui velocidades até 3 Mbit/s, a versão 3.0 High Speed (HS) que inclui o protocol Adaptation Layer do 802.11 permitindo taxas de transferência até 24 Mbit/s e a versão 4.0 Low Energy que permite velocidades até 1 Mbit/s com um consumo de energia muito baixo, adoptadas em Julho de 2007, Abril de 2009 e em Dezembro de 2009 por [57].

Este protocolo opera na faixa de frequência dos 2.4 GHz, podendo na sua versão 3.0 ser

configurado para operar na faixa dos 5 GHz. A versão 2.1 e 4.0 do Bluetooth utilizam uma frequência de saltos rápida de comutação de pacotes a fim de minimizar a interferência com outros produtos que utilizem a mesma faixa de frequências. Já a versão 3.0 utiliza o protocolo CSMA/CA com detecção de colisão para minimizar essas interferências. O Bluetooth suporta ligações ponto a ponto, bem como ligações ponto a multiponto. Esta tecnologia permite dois tipos de topologias de rede, a Piconet e a Scatternet. Uma Piconet é formada por um dispositivo como master e até 7 como escravos, permitindo a ligação total de até 8 dispositivos. Uma Scatternet é uma associação de duas ou mais Piconets sobrepostas no espaço e no tempo. Um dispositivo Bluetooth pode participar em várias Piconets ao mesmo tempo, permitindo que a informação possa fluir além da área de cobertura de uma simples Piconet. Um dispositivo numa Scaternet pode ser escravo em várias piconets, mas mestre só numa delas. O Bluetooth 4.0 tem uma topologia de rede "star-bus", o que permite a ligação de um grande número de dispositivos, mais concretamente 2^{48} dispositivos.

2.4.2 UWB (IEEE 802.15.3)

UWB [58, 59] é uma tecnologia de redes de área de pessoal sem fios (WPAN) que permite velocidades de transferência até 480Mbit/s com um alcance até 10 metros. A sua frequência de funcionamento pode ir desde os 3.1 GHz até aos 10.6 GHz, fazendo a reutilização do espectro já atribuído a outros serviços coexistindo com estes. Esta coexistência é feita à custa da distribuição do sinal a transmitir por uma grande largura de banda, sendo o sinal emitido gerado com níveis de potência muito baixa (níveis baixos de densidade espectral de potencia), ou seja, gera um sinal de muito grande espectro. Este sinal é gerado via espalhamento espectral de sequência directa (DSSS) utilizando chip-rates muito elevados ou via utilização de pulsos com uma duração muito curta IR (Impulse Radio) e modulação PPM (Pulse position Modulation). Assume-se que um sinal é UWB quando a largura de banda ocupada é muito grande quando comparada com a frequência central (maior do que 25% da frequência central, ou maior que 1,5 GHz de largura de banda). Foram propostas várias soluções para a camada física desta tecnologia, mas acabaram-se por se fundirem em apenas duas, a Multi-Band Orthogonal frequency-division multiplexing (MB-OFDM), e direct-sequence UWB (DS-UWB). A primeira solução divide o espectro em 13 sub-bandas, enquanto a segunda divide o espectro em 2 sub-bandas.

2.4.3 Zig-Bee (IEEE 802.15.4)

O Zig-Bee [58, 60–62] é uma tecnologia de área pessoal sem fios (WPAN), que contrapondo-se aos elevados débitos oferecidos por tecnologias como o Bluetooth, Wi-Fi e UWB, pretende associar a transmissão de dados sem fios a um reduzido consumo de energia com uma elevada flexibilidade. Este protocolo possui uma pilha protocolar de implementação simplificada que conduz a interfaces de baixo custo. O Zig-Bee suporta uma elevada densidade de nós na rede, 65535 dispositivos por cada coordenador, atingindo um alcance máximo de 10 metros. Esta tecnologia admite diferentes tipos de topologias de rede, estrela, mesh e árvore, permitindo o estabelecimento de redes de nós "Ad-hoc. Este protocolo pode operar em diferentes faixas de frequência, a faixa dos 2.4GHz, que com 16 canal pode obter taxas de transmissão até 250 kbps, a faixa dos 915MHz, que com 10 canais pode obter taxas de transmissão na ordem dos 40Kbps, e a faixa dos 868 MHz que com 1 canal pode obter taxas de transmissão na ordem dos 20Kbps. Hoje em dia já se encontram módulos Zig-Bee que permitem um alcance até 100 metros, com velocidades de 250kbps.

2.4.4 Wi-Fi (IEEE 802.11 a/b/g/n)

O Wi-Fi [58, 62] é uma tecnologia de área local sem fios (WLAN), que permite o acesso à internet com grandes velocidades e grandes alcances. Este tecnologia permite dois tipos de topologias de rede, a baseada em infra-estrutura, onde todos dispositivos são ligados a um Access Point, e a ligação computador-a-computador chamada rede "Ad-Hoc". Devido ao sucesso do Wi-Fi esta é uma tecnologia muito investigada, disponibilizando até ao momento 4 normas, entre elas, a norma A, que opera na faixa de frequência de 5 GHz e possuí velocidades até 54 Mbps com um alcance de até 10 metros, a norma B que opera na faixa dos 2.4 GHz e atinge velocidades de até 11 Mbps com um alcance de até 100 metros, a norma G que opera nos 2.4 GHz, com velocidades de até 54 Mbps com um alcance de até 100 metros e a norma N que opera na faixa de frequência dos 2.4 ou 5 GHz permitindo velocidades de até 600 Mbps.

2.4.5 Comparação das tecnologias sem fios

Depois de uma breve introdução e apresentação das quatro tecnologias de área pessoal sem fios, apresenta-se um estudo comparativo de alguns factores importantes para o sistema a implementar. Este estudo tem como objectivo a escolha do melhor protocolo a utilizar para os equipamentos remotos no sistema de domótica.

2.4.5.1 Consumo de energia

A autonomia de um dispositivo móvel é um ponto fulcral que se pode traduzir na aceitação ou não de um produto móvel. O consumo de energia, tal como a capacidade das baterias são dois dos factores que influenciam a autonomia de um dispositivo. Por isso, é sempre pretendido que o consumo de energia seja o mais baixo possível para que as soluções possam estar um longo período de tempo activas sem necessidade do recarregamento de baterias. Na tabela 2.1 são apresentados valores comparativos do gasto de energia das quatro tecnologias sem fio abordadas, usando chipsets específicos.

Standard	Bluetooth	UWB	ZigBee	Wi-Fi
Chipset	BlueCore 2	XS110	CC2430	CS53111
VDD (Volt)	1.8	3.3	3.0	3.3
Tx (mA)	57	~227.3	24.7	219
Rx (mA)	47	~227.3	27	215
Pt (mW)	187.2	~1500.2	155.1	1432.2
Bit Rate (Mbps)	0.72	114	0.25	54

Tabela 2.1: Comparativo do consumo de energia das tecnologias. Adaptada de [58].

Como se pode verificar pelos valores da tabela, as tecnologias UWB e Wi-Fi possuem um consumo de energia consideravelmente maior que o Bluetooth e Zig-Bee. Através da tabela verifica-se que a potência consumida pelas tecnologias Zig-Bee e Bluetooth, é consideravelmente baixa para os chipsets comparados. Em [58] pode ser visualizado um gráfico que mostra que a tecnologia UWB e Wi-Fi possuem uma melhor eficiência no consumo de energia, caso seja considerado um consumo de energia normalizado, baseado numa alta taxa de dados (mJ/MB), como por exemplo vídeo.

2.4.5.2 Tempo de transmissão

O valor do tempo de transmissão é dado pelo rácio entre o tamanho do pacote e a velocidade de transferência, ou seja, é dependente de velocidade de transmissão do canal por onde é transmitido e do seu tamanho. Na tabela 2.2, é possível verificar os valores típicos de cada um dos protocolos apresentados.

Standard	Bluetooth	UWB	ZigBee	Wi-Fi
IEEE Spec.	802.15.1	802.15.3	802.15.4	802.11a/b/g
Max Data Rate (Mbit/s)	0.72	110+	0.25	54
Bit time (uS)	1.39	0.009	4	0.0185
Max data payload (bytes)	339 (DH5)	2044	102	2312
Max Overhead (bytes)	158/5	42	31	58
Coding Efficiency* (%)	94.41	97.94	76.52	97.18
+ Unapproved 802.15.3a	*Data =	10Kbytes		

Tabela 2.2: Parâmetros típicos das tecnologias [58].

Através desta tabela, é possível concluir que o tempo de transmissão no protocolo Zig-Bee é o maior, devido à sua baixa velocidade de transmissão e que o UWB possui o tempo de transmissão mais rápido.

2.4.5.3 Eficiência da codificação de dados

A eficiência de codificação é dada pelo rácio entre o tamanho dos dados e o tamanho da mensagem a transmitir.

Dado que o máximo de dados úteis é de 339 para o Bluetooth, 2044 para o UWB, 102 para o ZigBee e 2312 para o Wi-Fi, é possível deduzir que o protocolo Bluetooth possui uma maior eficiência para dados em torno dos 339 Bytes, enquanto que o Zig-Bee possui boa eficiência para tamanhos de dados menores do que 102 Bytes. Como pode ser visualizado na tabela 2.2, para grandes quantidades de dados, o Bluetooth, UWB e Wi-Fi têm maior eficiência (superior a 94%), em comparação com os 76,52% do ZigBee (valores para dados de 10KBytes). Estes valores são explicados devido à necessidade da fragmentação de dados. Assim, é possível deduzir

que à medida que os dados vão aumentando, existirão maiores perdas de eficiência em todas as tecnologias, pois existirá a necessidade de fazer fragmentação de dados.

2.4.5.4 Coexistência de sinais entre tecnologias

O Bluetooth, Wi-Fi e Zig-Bee operam usualmente na mesma faixa de frequência (2.4 GHz). A utilização destes protocolos em simultâneo pode dar origem a interferências nos seus sinais, originando um decréscimo de qualidade de serviço. Este factor torna-se ainda mais relevante quando o Wi-Fi e o Bluetooth são tecnologias muito e cada vez mais utilizadas em pequenos dispositivos, como telemóveis.

Entre o protocolo Zig-bee e Wi-Fi 802.11g, o último é mais afectado em uplink do que em downlink pelo Zig-Bee, enquanto a tecnologia 802.11g afecta o desempenho do Zigbee quando o espectro do canal de operação entre eles coincide [63]. Por este motivo, é recomendado um deslocamento de 7 Mhz entre as frequências do 802.11 e 802.15.4 para um desempenho satisfatório do último [64].

Da mesma forma, o protocolo 802.11g é mais afectado pelo Bluetooth do que pelo Zig-Bee [63], existindo uma grande probabilidade de erros nos pacotes do protocolo 802.11g quando uma rede Bluetooth está activa [61]. No entanto estes erros podem ser atenuados significativamente através da utilização de técnicas de modelagem de tráfego de Bluetooth simples [65].

2.4.5.5 Sistemas de localização em tempo real

Os sistemas de localização em tempo real, tal como o nome indica, permitem identificar a localização de determinada pessoa ou objecto num determinado espaço. Actualmente existem diversos sistemas deste tipo, tendo cada um as suas próprias características. A acuidade do sistema de localização, o tempo de resposta e o tipo de localização fornecida (2D ou 3D) são os três factores mais importantes neste tipo de sistemas.

Dum estudo sobre sistemas de localização em tempo real [66–68], foi deduzida a tabela 2.3 onde constam os parâmetros mais importantes de algumas das tecnologias existentes.

Através da análise desta tabela, é possível verificar que a tecnologia UWB se destaca neste tipo de sistemas. Possui a acuidade mais baixa, os melhores tempos de resposta, e permite uma

Sistemas de Localização					
Sistema	**Tecnologia sem fios**	**Acuidade**	**Precisão**	**Complexibilidade**	**Dimensão**
Micosoft Radar	WLAN	2-3m	50% em torno dos 2.5m; 90% em torno dos 5.9m	Moderada	2D, 3D
DIT	WLAN	3m	50% em torno dos 5.12m através de SVM(support vector machine); 90% em torno dos 5.4m através de MLP (multi-layer perceptron)	Moderada	2D, 3D
Ubisense	UWB	15cm	99% em torno dos 0.3m	Resposta em tempore-al (1Hz-10Hz)	2D, 3D
Sappire Dart	UWB	<0.3m	50% em torno dos 0.3m	Frequência de respos-ta (0.1Hz - 10 Hz)	2D, 3D
Topaz	Bluetooth	2-3m	95% em torno dos 2m	Atraso de posiciona-mento 15 - 30 segun-dos	2D
Blip Systems BlipNet	Bluetooth	10cm-10m	-	Tempo de resposta superior a 5 segundos	2D

Tabela 2.3: Sistemas de localização. Deduzida de [66–68].

localização em três dimensões. Já na tecnologia Wi-Fi, os sistemas apresentados apresentam uma acuidade a rondar os 2.5 metros, com uma precisão baixa. Para a tecnologia Bluetooth é verificado que os sistemas de localização possuem um tempo de resposta relativamente elevado. Relativamente à tecnologia Zig-Bee não foi encontrado nenhum sistema de localização.

Por fim, a figura 2.4, vem confirmar que o UWB é a tecnologia que possui melhor acuidade relativamente a todos as tecnologias sem fios apresentadas.

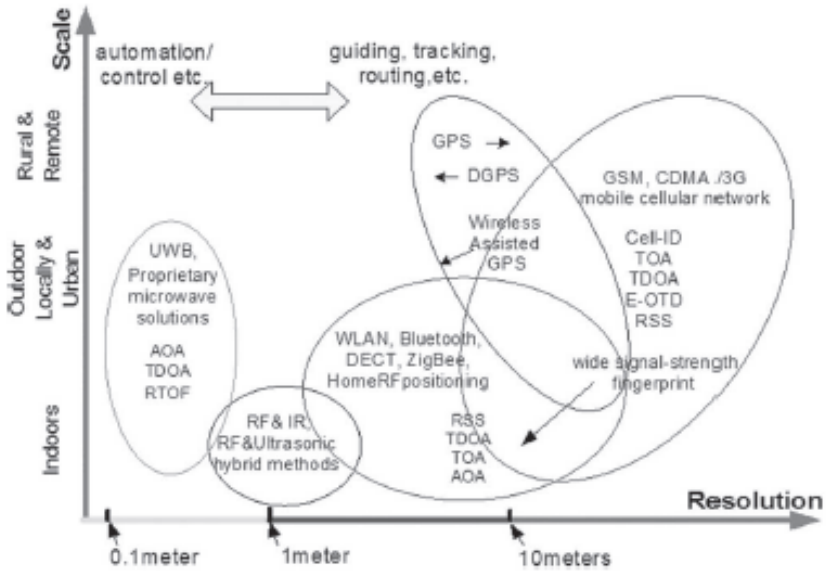


Figura 2.4: Comparação da capacidade de localização das tecnologias [67].

2.5 Protocolo SNMP

Nesta secção é feita uma abordagem ao protocolo SNMP [69], no que se refere à sua arquitectura, tipos de operações, e às suas principais vantagens e desvantagens no seio da domótica.

O SNMP é um protocolo assíncrono e assimétrico, não orientado à conexão e utiliza o protocolo de transporte User Datagram Protocol (UDP) para a troca de informações. Surgiu com a necessidade da criação de mecanismos que permitissem gerir a rede TCP/IP. Devido a sua facilidade de utilização e ao facto de se revelar uma ferramenta poderosa na gestão de redes heterogéneas fez com que a sua utilização fosse expandida para outro tipo de sistemas e equipamentos, como encaminhadores, comutadores, computadores, etc.

Esta expansão fez com que novas actualizações surgissem, e numa segunda fase foi lançada a sua segunda versão. Esta versão trouxe melhorias de performance, segurança, confidencialidade e acrescenta novas funcionalidades, como o aparecimento de novas primitivas operacionais (GetBulkRequest, InformRequest), juntamente com utilização da Structure of Managed Information v2 (SMIv2), onde são introduzidos novos tipos de dados. Mais tarde é lançada a última versão, o SNMPv3, que acrescenta a obrigatoriedade do uso de mecanismos de segurança robustos.

No protocolo SNMP existem duas entidades principais, que permitem a troca de informação: o gestor e agente SNMP. O gestor SNMP utiliza primitivas SNMP para solicitar ou modificar informação nas tabelas fornecidas pelo agentes, podendo receber notificações, sempre que alguma situação previamente configurada no agente ocorra. Por sua vez, o agente SNMP é a entidade que está presente no equipamento/sistema gerido, tendo como principal função a recepção e processamento de primitivas que provêm do gestor. Todos os comandos provenientes do gestor têm associado, pelo menos, um Object Identifier (OID), que refere um objecto num elemento fundamental da arquitectura SNMP, a MIB, que funciona como uma base de dados estruturada sob a forma de uma árvore onde estão armazenadas todas as informações de monitorização/configuração.

2.5.1 Primitivas SNMP

As primitivas SNMP permitem a consulta e alteração da informação de gestão presente na MIB e o aviso de acontecimentos previamente configurados. Para operações de consulta são disponibilizadas três primitivas:

- GET: permite recuperar uma instância de um objecto específico;
- GETNEXT: permite emitir uma sequência de comandos obtendo um grupo de valores da MIB seguindo uma ordem lexicográfica;
- GETBULK: definida apenas na segunda versão do SNMP, que permite recuperar um conjunto de informação presente numa tabela de uma só vez, através da utilização de dois campos nonrepeaters e max-repetitions. Nonrepeaters indica que os primeiros N objectos podem ser recuperados apenas com uma simples operação GETNEXT, enquanto que o campo Max-repetitions indica o número de tentativas da operação GETNEXT a serem realizadas para recuperar os demais objectos.

Para realizar uma operação de inserção ou alteração de um determinado campo existe a primitiva SET. Esta primitiva efectua também a criação de uma nova linha numa tabela, caso a mesma não exista. Para a notificação de eventos existem duas primitivas:

- TRAP: permite dar a conhecer a um agente SNMP uma determinada ocorrência
- INFORM: permite a troca de informação entre gestores.

2.5.2 MIB

A MIB é um dos componentes essenciais associados ao protocolo SNMP, pois permite a especificação e manipulação de todas as informações disponibilizadas pelo agente SNMP. Em cada MIB é especificado um conjunto de objectos, definidos usando a linguagem declarativa Abstract Syntax Notation (ASN.1). Os objectos são organizados através de uma árvore de dados, onde cada nó é identificado por um Object Identifier (OID) que corresponde a uma sequência de números separados por pontos que são gerados com base na estrutura hierárquica definida na criação dos objectos. Cada objecto definido pode ter até 3 tipos de acesso:

- "read-only": define que o seu valor só pode ser consultado;
- "read-write": define que o valor tem permissões de leitura e escrita;
- "read-create": Atribui privilégios de criação, leitura, escrita a um objecto.

A forma como os objectos são definidos na MIB encontra-se definida na Structure of Managed Information (SMI) e obedece à seguinte estrutura:

- Nome: identificador único do objecto (OID). O nome pode ser apresentado numericamente ou textualmente;
- Tipo e Sintaxe: o tipo de cada objecto é definido usando a linguagem ASN.1, que especifica a forma como os dados são representados e transmitidos entre agentes gestores.
- Codificação: cada objecto definido na MIB é codificado e descodificado seguindo a codificação Basic Encoding Rules (BER), que permite a correcta transmissão dos dados.

A SMI também define quais os tipos de dados permitidos nas MIBs. Podem assim ser utilizados:

- Integer, Integer32: número inteiro de 4 Bytes, que pode ter o seu alcance especificado explicitamente; O tipo Integer32 encontra-se definido na SMIv2;
- Integer64: tem a mesma funcionalidade do anterior, é definido na SMIv2 e tem a capacidade de 8 Bytes.
- Unsigned, Unsigned32: número inteiro de 4 Bytes não negativo, que pode ter o seu alcance especificado explicitamente. O tipo Unsigned32 encontra-se definido na SMIv2;
- Unsigned64: número de 8 Bytes não negativo, definido na SMIv2 que pode ter o seu alcance especificado explicitamente;
- enumerated: valores maiores que zero, atribuídos a um determinado objecto, fazendo com que este só possa tomar algum desses valores.
- Gauge e Gauge32: valor inteiro não-negativo de 4 Bytes, que pode ser incrementado e decrementado. O tipo Gauge32 surge na SMIv2;

- Gauge64: tem a mesma funcionalidade do anterior, é definido na SMIv2 e tem a capacidade de 8 Bytes.
- Counter e Counter32: valor inteiro não-negativo de 4 Bytes, que pode ser incrementado mas não decrementado. Quando o seu valor máximo é atingido, o contador recomeça a contagem do zero. O tipo Counter32 encontra-se definido na SMIv2;
- Counter64: tem a mesma funcionalidade do anterior, é definido na SMIv2 e tem uma capacidade de 8 Bytes.
- TimeTicks: especifica um valor inteiro não-negativo que conta o tempo em segundos;
- OCTET STRING: utilizado para especificar octetos de informação textual ou binária. Na SMIv1 não existe limite no número máximo de octetos, porém na SMIv2 pode ser estabelecido um limite máximo;
- OBJECT IDENTIFIER: tipo utilizado para identificar um único objecto, consistindo numa sequência de inteiros separados por pontos;
- IpAdress: tipo de dados utilizado para endereços do tipo ipv4.
- NetworkAddress: tipo de dados para endereços do tipo ipv4, apenas disponível na SMIv1 e utilizado para representar endereços de rede de outras famílias;
- Opaque: tipo de dados utilizado para especificar octetos de informação binária. Permite qualquer formato de dados quer definido na ASN.1 ou noutra sintaxe;
- BITS: definido apenas na SMIv2, serve para representar zero ou mais bits que de um valor;
- Sequence e Sequence of: estes tipos de dados são utilizados para a construção de tabelas.

2.5.3 Adequação protocolo SNMP em serviços de domótica

Em [70], é apresentado um estudo sobre o desempenho do protocolo SNMP, quando aplicado para monitorização e controlo de diferentes tipos de dispositivos de temperatura. Os resultados médios obtidos de "round-trip time" foram de 43 ms, o qual permitiu concluir que este protocolo possui um bom desempenho quando aplicado ao controlo e monitorização. Em [71], analisou-se a aplicabilidade do protocolo SNMP no contexto das redes de sensores sem fios Zig-Bee.

Os resultados obtidos confirmaram que o SNMP pode ser uma boa aposta para este tipo de aplicações, pela sua eficiência na transmissão capacidade de manipulação da informação e pelo facto de ser independente do tipo de sensores ou da tecnologia utilizada. Em [72] é apresentada uma solução para sistemas domóticos de baixo custo, flexíveis e escaláveis, baseadas numa tecnologia sem fios com comunicação através do protocolo SNMP. Através dos testes realizados foi concluído que o SNMP pode ser utilizado numa solução de domótica com tecnologias sem fios sem quaisquer restrições. Nos testes efectuados foi identificado um valor inferior a 20 ms entre pedido e resposta.

Em suma, o SNMP apresenta características que indiciam que a sua aplicação na domótica pode ser uma mais-valia, contudo, os trabalhos mencionados, levantam dúvidas na utilização deste protocolo em aplicações de tempo real. Torna-se assim necessário a realização de testes que permitam validar os serviços disponibilizados em domótica. É no entanto, constatável a sua grande simplicidade, segurança (SNMPv3), e redução de processamento e tráfego na rede fornecida pela capacidade do envio de notificações em situações programadas.

Assim, no contexto do sistema DOMinho, foi decidido utilizar o SNMP e as MIBs como tecnologia base para o desenvolvimento de um sistema modular e completamente normalizado para domótica.

2.6 Conclusões

2.6.1 Produtos comerciais

Após o estudo das informações disponibilizadas nos "websites" destas empresas e dos vários contactos estabelecidos, foi possível apurar, que todas dispõem de interfaces e serviços idênticos. Apesar da disponibilização de alguns serviços adaptativos, estes sistemas não possuem qualquer tipo de inferências sobre os utilizadores, sendo necessário uma configuração prévia destes serviços. As interfaces utilizadas são consideradas rígidas, pois não apresentam diferenças inerentes aos utilizadores. Deduz-se assim que este mercado ainda não está de todo sensibilizado para a utilização de interfaces inteligentes. Por outro lado, também os tipos de interacções fornecidas aos utilizadores nestes sistemas estão obsoletos, se comparados com os tipos de interacção naturais existentes, como a voz e os gestos.

É também constatável o uso de diferentes tecnologias sem fios para efectuar as comunicações entre os seus comandos remotos e os sistemas de domótica. A falta de normalização nos sistemas de domótica actuais é um factor negativo, tanto para empresas, como para potenciais clientes, a nível individual por não permitir a optimização de processos e resultados dos sistemas domóticos e a interoperabilidade entre eles. Dos serviços disponibilizados por estas empresas, salienta-se a adaptação da empresa Casa do Futuro, onde o sistema acende automaticamente as luzes da casa, activa a música preferida do utilizador e acciona o ar condicionado para o mesmo. Por sua vez o sistema Cardio possui também um tipo de adaptação no menu de "programação", no entanto, necessita da introdução de um código por parte do utilizador, o que não é de todo esperado num sistema inteligente.

Todos estes factos vêm realçar a sustentabilidade dum projecto do tipo DOMinho, com os seus requisitos de normalização, adaptação e generalidade que permitem a interoperabilidade entre sistemas.

2.6.2 Interfaces inteligentes

Após a análise das vantagens das interfaces de utilizador inteligentes, é possível concluir que a sua utilização vem aumentar a qualidade de vida dos seus utilizadores. As interfaces adaptativas ajudam e simplificam a vida dos utilizadores, podendo:

- omitir as opções não utilizadas;
- apresentar sugestões ao utilizador quando este inicia determinado serviço tendo em consideração as suas preferências;
- apresentar menus consoante os serviços e equipamentos que possui na localização onde está a ser acedido
- ter em consideração a presença de mais utilizadores no local, apresentando os menus preferidos por esse conjunto de utilizadores;
- omitir opções executadas por um certo utilizador quando se encontra na presença de outros.

As interacções inteligentes são úteis em diferentes situações, não existindo nenhum tipo especifico preferencial, pois, cada uma tem as suas próprias vantagens, sendo apropriadas consoante o contexto em que o utilizador se encontra.

2.6.3 Tecnologias sem fios

Após a análise comparativa das tecnologias sem fios apresentadas, foi concluído que nenhuma das tecnologias estudadas é plenamente adequada ao sistema pretendido.

Se, por um lado, pretende-se uma tecnologia sem fio com uma elevada largura de banda, de modo a que novos serviços possam ser adicionados sem limitações, como por exemplo, o streaming de vídeo de câmaras de segurança, o acesso à internet e a disponibilização de jogos, por outro lado, pretende-se que a tecnologia sem fio escolhida consuma pouca energia de modo a que o comando domótico possua uma grande autonomia. No entanto, foi verificado que o consumo de energia das tecnologias sem fio apresentadas é tanto maior, quanto maior é a sua largura de banda. Se for considerado a eficiência de codificação dos dados de um protocolo, este também interfere com o consumo de energia, pois, para a mesma quantidade de dados, pode ser necessária a fragmentação de dados. Esta fragmentação é apenas necessária se o número máximo de dados a enviar em cada mensagem, for superior ao número de dados disponível no payload na tecnologia utilizada. O número de dados para o serviço proposto, não é fixo, não podendo ser concluído qual o melhor protocolo a utilizar neste âmbito. No entanto, se considerado apenas o serviço de gestão de interfaces, a quantidade de dados de payload deverá rondar os 300 Bytes (valores médios de capturas realizadas ao serviço desenvolvido). Tendo em consideração este valor, o protocolo Bluetooth seria o protocolo melhor adaptado.

Quando à coexistência de sinais entre tecnologias sem fio, o UWB leva vantagem em relação aos protocolos comparados, pois estes funcionam na faixa de frequência dos 2.4 GHz, enquanto o UWB utiliza o espectro entre os 3.1GHz e os 10.6 Ghz. Através da utilização desta faixa de frequência, este protocolo não interfere com o protocolo 802.11g e Bluetooth, largamente utilizados para acesso à internet ou transferências de dados entre telemóveis.

Por fim, considerando os sistemas de localização em tempo real existentes, também o UWB leva vantagem. Este protocolo possui a melhor acuidade/precisão, o melhor tempo de resposta, e permite a localização em três dimensões. Deste modo, a tecnologia UWB deverá ser a escolhida

para o sistema DOMinho. Apesar da mesma não possuir um baixo consumo de energia, este factor pode ser atenuado através da utilização de baterias com uma grande eficiência energética. A eficiência da codificação de dados pode também não ser a mais adequada para a comunicação SNMP, no entanto, pode trazer vantagens para novos serviços. Em todas as outras características, este protocolo leva vantagem sobre os restantes comparados.

Capítulo 3

Definição do sistema

3.1 Projecto DOMinho

O trabalho descrito nesta dissertação insere-se no âmbito do projecto DOMinho [2]. O objectivo deste projecto consiste na criação de um sistema domótico inteligente, genérico, modular e normalizado.

Na Figura 3.1 são apresentadas as diferentes camadas constituintes deste projecto, tendo já sido apresentado na figura 1.1 a sua arquitectura geral.

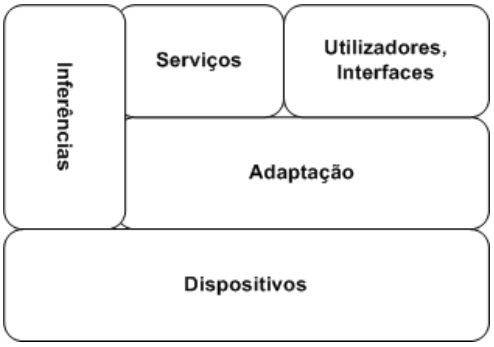


Figura 3.1: Camadas do projecto DOMinho.

Na camada de mais baixo nível deste sistema, encontram-se todos os dispositivos responsáveis pela captura de informação e execução de acções sobre o ambiente em que se encontram inseridos. Através da camada de Adaptação disponibilizada pelos equipamentos e "gateways", todas as informações destes dispositivos são convertidas em objectos de MIBs, tornando possível a comunicação com a camada de serviços através de SNMP.

A camada Serviços controla todos os serviços domóticos disponibilizados na habitação (iluminação, climatização, multimédia, etc). Esta camada é também responsável pela consulta e alteração de informações contidas nas MIBs da camada de Adaptação.

Ao mesmo nível da camada de serviços é disponibilizado o módulo de interfaces e utilizadores. O primeiro módulo permite a comunicação entre utilizadores e o sistema DOMinho. Este módulo, genérico, permite que informações, layouts e modos de interacção possam ser adaptados aos utilizadores. De forma a que estas adaptações sejam realizadas, existe a necessidade do conhecimento das preferências, hábitos e comportamentos dos utilizadores. Para tal, foi criado o módulo de utilizadores que disponibiliza informações dinâmicas e dependentes de contextos que permitem que as adaptações de interfaces e serviços sejam mais precisas.

A camada de inferências é responsável por conferir ”inteligência”ao sistema, recorrendo à utilização de ontologias e às informações disponibilizadas pelo módulo de utilizadores, para a gestão de serviços e a criação de interfaces adaptados aos utilizadores.

3.2 Modelo de utilizadores e interfaces

O modelo de utilizadores e interfaces é composto por dois componentes, o módulo de utilizadores e o módulo de interfaces. No contexto desta dissertação, além da implementação destes dois componentes foi implementado um protótipo de uma interface gráfica que tem como objectivo validar os componentes desenvolvidos. Podemos ver na 3.2 uma representação destes componentes.

3.3 Módulo de utilizadores

O componente de utilizadores proposto permite armazenar informações relativas aos utilizadores no sistema DOMinho. Estas informações permitem suportar o conhecimento das preferências, hábitos e comportamentos de cada utilizador, podendo ser utilizadas pelos softwares inteligentes para inferir interfaces e gerir serviços de forma autónoma. É proposto que estas informações sejam contextualizadas, sendo considerados cinco contextos diferentes:

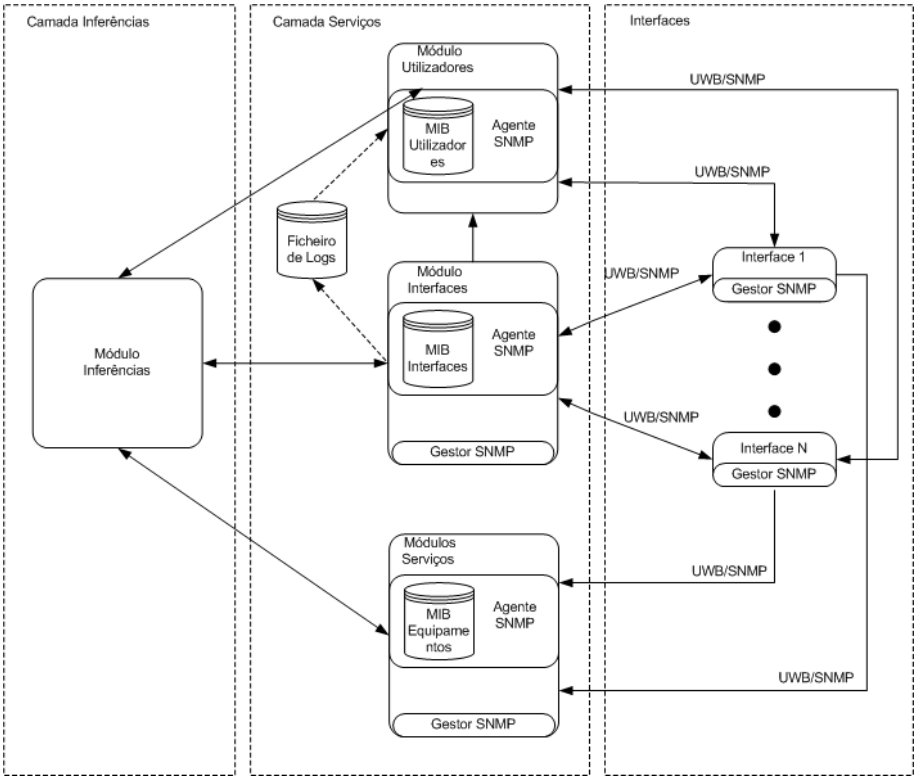


Figura 3.2: Arquitectura do sistema proposto.

- Contexto temporal: permite que possam ser encontrados padrões temporais nas preferências e comportamentos dos utilizadores;
- Contexto espacial: complementa o contexto temporal, permitindo que possa ser conhecido o local onde as acções são executadas;
- Contexto de utilizador: neste sistema não é esquecido o conceito de privacidade, ou seja, sempre que determinada acção é realizada, é guardada a informação de todos os utilizadores presentes no local. Este contexto torna possível que o sistema conheça as preferências dos utilizadores quando estes estão acompanhados, ou seja, torna possível a identificação dos serviços preferidos para um agregado de utilizadores;
- Contexto de hardware: permite que as preferências dos utilizadores para um determinado hardware sejam conhecidas, fazendo com que possam ser reflectidas automaticamente na gestão dos serviços disponibilizados;
- Contexto de interface: permite conhecer as interacções realizadas por um dado utilizador através de uma interface. Assim, é possível inferir, por exemplo, que o utilizador X só

utiliza a interface Y para controlar a temperatura, logo a interface pode ser adaptada tendo em consideração esta informação.

Com estes contextos o sistema consegue inferir as diferenças inerentes aos utilizadores. No entanto, para que possam ser úteis, é necessário conhecer os comportamentos dos utilizadores ao longo do tempo. Para tal, é proposto que este sistema seja dinâmico, ou seja, todas as acções realizadas pelos utilizadores no sistema sejam armazenadas em base de dados (MIB) pelo sistema de interfaces. Esta base de dados é periodicamente acedido pelo sistema de utilizadores que calcula estatísticas com essas informações. As estatísticas têm em consideração os contextos referidos e são guardadas numa tabela de estatísticas. As entradas desta tabela podem posteriormente ser acedidas pelos módulos de inferências, de modo a detectarem possíveis alterações nas preferências dos utilizadores.

Este mecanismo facilita o armazenamento das informações inferidas pelos softwares inteligentes. Deste modo, foram moduladas duas tabelas onde podem ser registadas essas inferências. Por outro lado, também a tabela de perfis de utilizador tem noção dos contextos referidos. Através desta associação é possível que os softwares inteligentes possam inferir e registar diferenças nos perfis de utilizador consoante o contexto no qual se encontrem.

Por fim, e como é impossível conhecer um novo utilizador quando este se regista pela primeira vez no sistema, é pretendido que este possa ser caracterizado globalmente. Para tal, é proposto a utilização de estereótipos, que têm como função atribuir um perfil de utilizador médio. Assim, é possível ao sistema arrancar com um conhecimento prévio de cada utilizador, desde o seu registo.

De frisar ainda que as tabelas de estatísticas, inferências, perfis de utilizador, estereótipos e inferências de favoritos moduladas, têm associadas um campo, onde pode ser reflectido o nível de confiança associado a cada entrada. Este valor foi baseado no projecto Amigo [9], e permite atribuir um factor diferencial aos valores armazenados.

3.4 Módulo de interfaces

O módulo de interfaces proposto permite a comunicação entre equipamentos interface e o sistema de domótica, sendo permitido que tanto o utilizador como o sistema possam iniciar a comunicação.

Através deste módulo é possível a utilização de interfaces básicos e interfaces complexos, ou seja, além de ser permitido definir todos os conteúdos de apresentação aos utilizadores, é permitido definir todo a formatação de saída da interface. Para tal, a interface necessita de identificar qual a sua complexibilidade, bastando preencher a tabela de funcionalidades e características disponibilizadas.

Este componente possibilita também a existência dum tipo de interacção com o utilizador e permite que novas actividades, características e funcionalidades possam ser adicionados sem qualquer reestruturação do mesmo.

Este módulo possui também um importante papel na obtenção do modelo de utilizador dinâmico, armazenando todas as interacções realizadas pelos utilizadores no sistema. Estas interacções são guardadas em MIBs, juntamente com a informação dos contextos em que foram executadas.

3.5 Protótipo

O protótipo desenvolvido pretende avaliar as tabelas moduladas, a validade científica e eventual utilidade comercial da solução proposta. A construção do protótipo, além do módulo de utilizadores e interfaces, inclui o desenvolvimento de uma interface gráfica que se adapta ao utilizador, através da utilização das informações fornecidas pelo módulo de interfaces. Neste protótipo, apenas é suportado a adaptação de conteúdos, não existindo nenhuma funcionalidade que permita a alteração do seu "layout" de saída.

Através do módulo de utilizadores, pretende-se a realização do registo e login dos utilizadores (com o envio da resposta de autenticação através de notificação), a leitura de um ficheiro de estereótipos e o preenchimento da tabela de perfis de utilizador baseado no estereótipo no qual é inserido o utilizador aquando do seu registo. Do mesmo modo, pretende-se executar uma função periódica para o registo de estatísticas, baseada nas informações das interacções realizadas pelos utilizadores.

Através do módulo de interfaces, pretende-se que todas as interfaces possam interagir com o sistema DOMinho. Além da disponibilização das informações presentes nas suas tabelas, pretende-se validar o envio das notificações definidas na MIB e a geração de um ficheiro com

as interacções realizadas pelos utilizadores.

3.6 Análise do sistema

O sistema de utilizadores e interface propostos foram desenvolvidos de forma genérica, o que permite que novas informações possam ser adicionadas sem necessidade de uma reestruturação do sistema.

De forma a normalizar e uniformizar todo o sistema de domótica foi criada uma estrutura genérica baseada em tabelas para uso comum. Esta estrutura, foi mapeada numa grande MIB, que pode ser consultada em [2]. De forma a poder ser testado o sistema desenvolvido, foram iniciados alguns valores para a estrutura criada. No entanto, de modo a que todas as informações do sistema sejam constantemente actualizadas, pretende-se que esta estrutura seja gerida por uma entidade autónoma, denominada genericamente de "comunidade de domótica".

As secções seguintes descrevem os componentes modulados para a gestão de utilizadores e interfaces propostos para o projecto DOMinho, bem como a estrutura criada para a "comunidade de domótica" dos módulos referidos.

3.7 Comunidade de domótica

A "comunidade de domótica" reflecte uma entidade virtual representativa de todos os indivíduos, empresas e instituições que contribuem para a proposta, discussão e definição de um conjunto de informações e valores normalizados (de uso universal). Assim, foram moduladas diferentes tabelas para esta comunidade, de forma a normalizar o sistema de utilizadores e interfaces desenvolvidos no contexto DOMinho.

3.7.1 Módulo de utilizadores

Para o sistema de utilizadores foram moduladas 4 tabelas.

3.7.1.1 Tabela perfil de utilizador

A tabela de perfis de utilizadores representada em 3.1 define uma estrutura que permite albergar de forma hierárquica todos os perfis de utilizador definidos para o sistema, assim como os valores que lhes podem ser atribuídos. Todos os valores apresentados na tabela são meramente exemplificativos, e todos os campos preenchidos com o valor zero significam que o seu atributo não é utilizado na entrada respectiva.

PerfilUtilizador				
Index(I)	**NomePerfil(S)**	**idxTipoValorPossível (S)**	**OIDValorPossível (S)**	**idxPerfilsUtilizadorPai(i)**
1	Preferências	0	0	0
2	Detalhes Pessoais	0	0	0
3	Nome	1	0	2
4	Multimédia	3	1.3.6.1.3.70.2.2.1.1.1	1
5	Filmes	3	1.3.6.1.3.70.2.2.1.2.1	4
6	Género Filmes	3	1.3.6.1.3.70.2.2.1.3.1	5

Tabela 3.1: Tabela da comunidade de perfis de utilizador.

Esta tabela possui 5 atributos, os quais permitem o delineamento dos perfis de utilizador de forma genérica.

- index: identificador único de cada perfil.
- nomePerfil: identifica o perfil.
- idxTipoValorPossível: contem informação do index da tabela "TipoValorPossivel", onde deve ser consultado qual o tipo de valor (Ex: único, lista sequencial, tuplo) definido para o perfil. Tendo em consideração a tabela apresentada, o perfil "Multimédia" foi definido com o valor 3. Deste modo, considerando os valores dados atribuídos à tabela de "TipoValorPossivel" conclui-se que o valor é definido como sendo do tipo tuplo, ou seja, este perfil apenas pode tomar um valor, sendo definido um limite mínimo, máximo e um valor que define qual a precisão que os valores atribuídos ao perfil podem tomar.
- OIDValorPossível: este campo contem a informação do OID da tabela onde podem ser encontrados os valores possíveis para o perfil definido.

- idxPerfilsUtilizadorPai: este campo permite a hierarquização de todos os perfis inseridos na tabela, sendo para tal, indicado qual o index do perfil "pai" da entrada em questão. Como pode ser verificado na tabela, o perfil "Género Filmes" é "filho" do perfil "filmes".

3.7.1.2 Tabela estereótipos comunidade

A tabela 3.2 permite a definição dos estereótipos possíveis para o sistema DOMinho. Tal como já foi referido, os estereótipos têm associado um conjunto de preferências que podem descrever um certo grupo de utilizadores. Quanto maior for o número de estereótipos criados maior é a probabilidade destes possuírem valores mais próximos das preferências reais dos utilizadores. Devido a enorme diferença das preferências dos utilizadores, se consideradas culturas, locais e Países, nesta tabela são somente definidos os estereótipos, deixando a responsabilidade da definição de valores mais específicos às comunidades nacionais, regionais e locais.

Estereótipos			
Index(I)	**Nome(S)**	**IdxEsteriotiposPai(I)**	**Descrição(S)**
1	Adulto	0	dependente do País
2	Homem	1	Sexo: Masculino
3	Mulher	1	Sexo: Feminino
4	Criança	0	dependente do País

Tabela 3.2: Tabela da comunidade de estereótipos.

Desta forma, foram definidos os seguintes atributos:

- index: identificador único de cada estereótipo.
- nome: identifica o estereótipo.
- idxEsteriotipoPai: permite a hierarquização de todos os estereótipos definidos na tabela.
- descrição: permite a inserção de uma breve descrição do estereótipos definidos, bem como a condição para o mesmo ser utilizado. No entanto, existem estereótipos onde a sua condição apenas pode ser definida nos sistemas de utilizador, pois pode variar consoante o País onde o sistema se encontra. Um exemplo deste tipo, é a definição de adulto, dado que a maioridade é atingida em diferentes idades se considerados diferentes Países.

3.7.1.3 Tabela tipo de favorito

De forma a se poder distinguir qual a preferência dos utilizadores para determinados serviços, foi criada a tabela 3.3. Através da sua utilização são normalizados determinados temas correspondentes aos serviços disponibilizados. Os valores das preferências do utilizador para cada um dos temas definidos podem assim ser inferidos pelos softwares inteligentes e enviados às interfaces para serem apresentados ao utilizador.

TipoFavorito	
Index(I)	**Favorito(S)**
1	Canal
2	Filme
3	Programa Favorito
4	Documentário
5	Música

Tabela 3.3: Tabela da comunidade de tipo favorito.

Esta tabela é formada por 2 atributos onde:

- index: identificador único de cada favorito.
- tipoFavorito: permite a definição de favoritos para diferentes serviços, como por exemplo, a definição de canal, de documentário e de programa para o serviço de televisão.

3.7.1.4 Tabela tipo de utilizador

A tabela 3.4 permite a definição de todos os tipos de utilizador que podem utilizar o sistema. Cada tipo de utilizador pode ter diferentes restrições a serviços e/ou a funcionalidades do sistema de domótica.

Esta tabela é formada por dois atributos:

- index: identificador único de cada Tipo de Utilizador.

TipoUtilizador	
Index(I)	**TipoUtilizador(S)**
1	Administrador
2	Utilizador Standard
3	Convidado

Tabela 3.4: Tabela da comunidade de tipo utilizador.

- tipoUtilizador: permite a definição de todos os tipos de utilizador disponíveis para o sistema. Neste caso, o tipo de utilizador administrador definido possui todos os privilégios do sistema. Os restantes, podem ter associadas restrições, sendo estas definidas em cada sistema de domótica.

3.7.2 Módulo de interfaces

Para o sistema de interfaces apenas foram criadas duas tabelas para a comunidade. Apesar de muito simples, as tabelas criadas permitem normalizar os tipos de menus e interacções que as interfaces podem utilizar para interagir com os utilizadores.

3.7.2.1 Tabela tipo de interacção

A tabela 3.5 permite a normalização de todos os tipos de interacção do sistema. Neste caso, e como referido em 2.6.2 é pretendido que todos os tipos de interacção existentes sejam definidos, pois, existem formas de interacção melhores do que outras dependendo da situação em que o utilizador se encontra.

TipoInteracao	
Índex(I)	**TipoInteracao(S)**
1	Touch
2	Áudio
3	Gráfica

Tabela 3.5: Tabela da comunidade de tipo interacção.

Os atributos desta tabela são:

- index: identificador único de cada tipo de interacção.
- tipoInteracao: este campo permite a identificação do tipo de interacção. Apesar de ser pretendido que todos os tipos de interacção sejam possíveis, esta tabela permite a sua normalização sendo simples a adição de novas formas de interacção.

3.7.2.2 Tabela tipo de menu

A tabela 3.6 permite a normalização de todos os tipos de menus que podem ser apresentados ao utilizador pelo equipamento interface. Esta normalização pretende que o interface apresente menus de diferentes formas ao utilizador, tendo como factor diferencial o seu tipo.

TipoMenu	
Index(I)	**Menu(S)**
1	Actividades
2	Aviso
3	Erro
4	Informação

Tabela 3.6: Tabela da comunidade de tipo menu.

Os atributos desta tabela são:

- index: identificador único de cada entrada na tabela.
- menu: neste campo, o tipo de menu “Actividades” indica que deve ser apresentado um menu requisitado pelo utilizador. O menu “aviso” indica que o sistema detectou algum problema. O menu “erro” indica que o sistema detectou um erro, e por fim, o menu “informação” indica que o sistema pretende dar alguma informação ao utilizador.

3.8 Módulo de utilizadores

Este módulo foi criado de modo a que todas as informações caracterizantes dos utilizadores possam ser armazenadas. É assim disponibilizado um sistema normalizado, dependente de contextos, que suporta informações dinâmicas acerca dos comportamentos dos utilizadores e que permite o acesso dessas informações a softwares inteligentes. Cada uma das tabelas moduladas possui também um campo de status, que permite que as entradas sejam dinamicamente removidas. Desta forma, sempre que seja inserido o valor "2" neste campo, a aplicação de utilizadores apagará a entrada respectiva.

Este módulo possui ainda três tabelas que têm como função distinguir três tipos de valores, inteiros, reais e Strings. Estas tabelas são utilizadas nos campos de "valor" das diferentes tabelas moduladas, de forma a que as aplicações possam identificar qual o tipo de valor definido.

3.8.1 Tabela autenticação

A tabela 3.7 permite que os utilizadores possam ser autenticados no sistema de domótica. Deste modo, sempre que um registo ou login é feito, é criada automaticamente uma nova entrada na tabela de utilizadores, onde são inseridos o "username" e "password" do utilizador. O resultado desta operação é dado pelo campo "valor".

Autenticacao			
Index(I)	**UserName(S)**	**Password(S)**	**Valor(S)**
1	Maria	Maria	R
2	Jose	Jose	L

Tabela 3.7: Tabela de autenticação.

Este tabela contem os seguintes atributos:

- index: identificador único de cada autenticação.
- username: Neste campo é recebido o username do utilizador.

- password: password do utilizador recebida da interface.
- valor: este campo permite a recepção do código da operação pretendida, ou seja, um "R" para um novo registo ou um "L" para login. Após o processamento da informação recebida, este valor é alterado, gerando uma notificação com este valor para a interface com a resposta à autenticação.

O valor enviado para o interface pode ter até três significados:

- maior do que 0: index da tabela utilizadores onde se encontra registado no utilizador;
- 0: Utilizador já registado em caso de registo, e password inválida em caso de login;

3.8.2 Tabela utilizadores

A tabela 3.8 permite o registo e acesso a informação básica de todos os utilizadores do sistema.

Utilizador						
Index(I)	**Nome(S)**	**IdxTipoUtilizador(I)**	**OIDLocalizacao(S)**	**Vc(I)**	**Vl(I)**	**Va(I)**
1	José	1	OID:ip:porta	23	5	4
2	Maria	2	OID:ip:porta	45	36	64

Tabela 3.8: Tabela de utilizadores.

Esta tabela contem os seguintes atributos:

- index: identificador único de cada utilizador.
- NomeUtilizador: Neste campo são definidos os nomes dos utilizadores registados no sistema.
- idxTipoUtilizador: identifica o tipo de utilizador. Os tipos de utilizador são definidos pela comunidade.
- OIDLocalizacao: identifica a localização do utilizador num espaço conhecido pelo sistema. Estes espaços encontram-se definidos numa tabela do sistema de serviços, e são calculados em função das coordenadas vc, vi e vl definidas nesta mesma tabela.

- Vc, Vi, Vl: Permite a inserção de uma coordenada tridimensional na tabela do utilizador, de modo ser conhecida a localização precisa do utilizador no sistema de domótica.

3.8.3 Tabela perfil de utilizador

A tabela 3.9 tem como objectivo permitir o armazenamento dos valores dos perfis definidos pela comunidade de domótica. Esta tabela permite também a associação de contextos a diferentes perfis de utilizador, de modo a possibilitar o conhecimento das preferências do utilizador quando estes se encontram em diferentes situações.

Perfil_Utilizador					
Index(I)	**IdxUtilizador(I)**	**IdxPerfilUtilizador(I)**	**OIDValor(VariablePointer)**	**NivelConfiança(I)**	**IdxContexto(I)**
1	1	4	OID	50	0
2	2	5	OID	34	1

Tabela 3.9: Tabela de perfil de utilizador.

Esta tabela foi modulada com os seguintes atributos:

- index: identificador único de cada perfil de utilizador
- idxUtilizador: identifica o index da tabela de utilizadores, referente ao utilizador visado pelo perfil.
- idxPerfilUtilizador: identifica o index do perfil de utilizador definido pela comunidade.
- OIDValor: este campo identifica o OID da tabela que contem o real valor associado ao perfil definido. A existência deste apontador deve-se à possibilidade da inserção de diferentes tipos de variáveis neste campo. Foram assim, criadas tabelas para valores do tipo real, inteiro e caracteres ascii (String). Assim, é possível às aplicações identificar qual o tipo de variável que podem encontrar.
- NivelConfiança: este campo permite que seja atribuído um nível de confiança ao valor do perfil.
- IdxContexto: identificador do índice da tabela contexto, onde são definidos os contextos para os quais o valor inserido se aplica.

3.8.4 Tabela estereótipos

Esta tabela permite que o sistema conheça de antemão um perfil básico para cada um dos estereótipos estipulados pela comunidade de domótica. Através da utilização desta informação, é armazenado um conhecimento sobre as preferências dos novos utilizadores, tornando possível futuras adaptações depois do registo de cada utilizador.

Deve ser acedida de forma automática pelo sistema de utilizadores quando um utilizador se regista no sistema. Para tal, cada novo registo deverá solicitar ao utilizador um número de dados pessoais suficientes que permita a verificação do estereótipo adequado para o utilizador.

Estereótipos				
Index(I)	**IdxEsteriotipo(I)**	**IdxPerfilUtilizador(I)**	**OIDValor(VariablePointer)**	**NivelConfiança(I)**
1	2	1	OID	100

Tabela 3.10: Tabela de estereótipos.

Os atributos definidos para esta tabela são:

- index: identificador único de cada estereótipo
- IdxEsteriotipo: identifica o index do estereótipo definido pela comunidade.
- IdxPerfilUtilizador: identifica o index do perfil de utilizador definido pela comunidade.
- OIDValor: identifica o OID da tabela onde pode ser encontrado o real valor definido para o estereótipo.
- NivelConfiança: identifica o nível de confiança associado ao valor atribuído ao estereótipo.

3.8.5 Tabela estatísticas

A tabela 3.11 permite a atribuição de um valor de confiança às interacções dos utilizadores no sistema de domótica. Foi modulada de forma a que cada interacção possa ser identificada tendo em consideração o contexto na qual foi efectuada, suportando um conhecimento das preferências

de cada utilizador quando estes se encontram em diferentes contextos. Para cada entrada desta tabela, é atribuído um nível de confiança que permite identificar quais os serviços mais utilizados pelos utilizadores num determinado contexto.

EstatisticasActividadesUtilizador					
Index(I)	**IdxActividade(I)**	**OIDValor(VariablePointer)**	**IdxUtilizador(I)**	**IdxContexto(I)**	**NivelConfiança(I)**
1	1	OID	1	1	75
2					

Tabela 3.11: Tabela de estatísticas.

Desta forma, os atributos moduladas para esta tabela foram:

- index: identificador único de cada entrada na tabela.
- idxActividade: identifica a actividade realizada pelo utilizador.
- OIDValor: este campo permite a identificação do OID onde pode ser visualizado o valor definido para a actividade realizada pelo utilizador. Por exemplo, para a actividade "mudar canal", este campo possui o valor do canal escolhido.
- idxUtilizador: identifica o utilizador que realizou a actividade.
- idxContexto: identifica o índice da tabela contexto, onde são definidos os contextos associados à actividade realizada.
- nivelConfiança: este campo permite a atribuição de um nível de confiança dado à actividade, ao valor definido e ao contexto na qual foi realizada.

3.8.6 Tabela inferências de utilizador

A tabela 3.12 foi modulada de forma a que os softwares inteligentes possam armazenar as informações das suas próprias inferências. Esta tabela possui os mesmos atributos da anterior e pode ser utilizada como um complemento às estatísticas efectuadas pelo agente de utilizadores, uma vez que as inferências podem extrair informações mais concretas das estatísticas efectuadas.

InferenciasActividadesUtilizador					
Index(I)	**IdxActividade(S)**	**OIDValor(VariablePointer)**	**IdxUtilizador(I)**	**IdxContexto(I)**	**NivelConfiança(I)**
1	1	OID	1	1	75
2					

Tabela 3.12: Tabela de Inferências de utilizador.

3.8.7 Tabela contextos

A tabela 3.13 permite a definição dos contextos propostos para o sistema de utilizadores. Pode ser atribuído um factor diferencial às informações retidas dos utilizadores.

Contextos					
Index(I)	**IdxContextoTemporal(I)**	**ContextoUtilizador(S)**	**OIDLocalizacao(S)**	**OIDEquipamento(S)**	**OIDInterface(S)**
1	1	1:2	OID.2:ip:porta	OID.1:ip:porta	OID.1:ip:porta
2	2	1	OID.2:ip:porta	0	OID.2:ip:porta

Tabela 3.13: Tabela de contextos.

Esta tabela possui os seguintes atributos:

- index: identificador único de cada perfil de utilizador
- idxContextoTemporal: identifica qual a entrada da tabela "contexto temporal" onde se encontra definido o contexto temporal associado a este contexto.
- contextoUtilizador: permite a identificação dos utilizadores que se encontram no mesmo espaço onde determinada acção foi tomada. Caso este campo possua mais do que um valor, é utilizado o carácter ":" para realizar a separação dos identificadores de cada utilizador. Caso o utilizador que interagiu com o sistema se encontrar sozinho, este campo deve ser preenchido com o valor correspondente ao seu próprio index de utilizador. Caso o sistema não possua qualquer tipo de sistema que permita identificar os utilizadores no sistema, este campo deve ser preenchido com o valor "0";
- OIDLocalizacao: permite conhecer a localização do utilizador. Este campo é preenchido com base no OID da tabela de localização do sistema de serviços.

- OIDEquipamento: permite identificar o equipamento utilizado pelo utilizador. Este campo é preenchido com o OID da tabela de equipamento do sistema de serviços. Caso o utilizador não esteja a interagir directamente com nenhum equipamento, este valor deve ser preenchido com o valor "0".
- OIDInterface: identifica o OID da tabela "interfaces" correspondente à interface utilizada pelo utilizador para realizar a actividade.

3.8.8 Tabela contexto temporal

A tabela 3.14 permite que sejam guardados os valores relativos ao espaço temporal em que foi efectuada determinada acção no sistema. Do mesmo modo pode ser utilizada para identificar um contexto temporal para uma determinada preferência do utilizador.

Para a definição desta tabela foram utilizados 10 atributos:

Contexto Temporal									
Index(I)	**DataHoraInicio(S)**	**DataHoraFim(S)**	**Segunda-feira(I)**	**Terça-feira(I)**	**Quarta-feira(I)**	**Quinta-feira(I)**	**Sexta-feira(I)**	**Sábado (I)**	**Domingo (I)**
1	2010-8-3,13:30:15.0	2010-8-13,13:30:15.0	1	1	1	1	1	2	1
2	2010-8-3,13:30:15.0	2010-8-13,13:30:15.0	1	1	1	1	1	1	1

Tabela 3.14: Tabela de contexto temporal.

- index: identificador único de cada contexto temporal.
- DataHoraInicio: identifica a data e hora de início de uma determinada acção ou da atribuição de uma dada preferência.
- DataHoraFim: identifica a data e hora em que finalizou a acção ou a preferência definida.
- Segunda-feira, Terça-feira, Quarta-feira, Quinta-feira, Sexta-feira, Sábado, Domingo: com estes campos permite-se identificar uma acção ou preferência que é válida para esse determinado dia da semana. Por exemplo, caso o atributo "sábado" e "domingo" sejam inseridos com o valor "2", deve-se ler que em todos os sábados e domingos correspondentes ao período de tempo decorridos entre os valores definidos nos campos "DataHoraFim" e "DataHoraInicio" são válidos para o contexto temporal definido.

3.8.9 Tabela inferências favoritos

Esta tabela permite o armazenamento das preferências do utilizador inferidas sobre determinado tema. É também possível, que o próprio utilizador altere e insira valores nesta tabela, respondendo a inquéritos/perguntas explicitamente implementadas nas interfaces.

InferenciasFavoritos					
Index(I)	**IdxTipoFavorito(S)**	**idxUtilizador(I)**	**idxContexto(I)**	**OIDValor(VariablePointer)**	**NivelConfiança(I)**
1	1	1	1	OID	60
2	2	1	2	OID	90

Tabela 3.15: Tabela inferências de favoritos.

Para a definição desta tabela foram utilizados os seguintes atributos:

- index: identificador único de cada entrada na tabela.
- idxTipoFavorito: este campo permite identificar qual o favorito a que a entrada se refere. Estes valores são definidos pela comunidade de domótica.
- idxUtilizador: identifica o utilizador para o qual o favorito foi inferido.
- idxContexto: identifica o contexto para o qual este favorito é válido. Caso este campo não seja pretendido deve ser colocado com o valor “0”.
- OIDValor: este campo identifica qual o valor associado ao favorito inserido. Por exemplo, para o “IdxTipoFavorito” com o valor “1”, correspondente ao favorito Canal, pode ser dado o valor “RTP1”. Este valor é mapeado na tabela do tipo Strings, sendo colocado neste campo o OID correspondente à sua entrada.
- NivelConfiança: identifica qual o nível de confiança associada ao valor do favorito definido.

3.9 Módulo de interfaces

O módulo de interfaces serve de gateway entre as interfaces e todo o sistema DOMinho. Para tal foram moduladas uma série de tabelas com requisitos genéricos que permitem a interacção com o sistema de domótica.

Tal como no sistema de utilizadores, todas as tabelas deste sistema possuem um campo de status de forma a que os diferentes módulos possam apagar entradas. Possui também três tabelas que têm como função distinguir três tipos de valores, inteiros, reais e Strings, que permitem que determinados campos possam assumir valores de diferentes tipos.

3.9.1 Tabela interfaces

Esta tabela permite o registo de todas as interfaces no sistema. Neste registo são guardados os valores básicos de cada interface, sendo permitido um rápido acesso a estes valores. Sempre que um novo interface acede ao sistema tem de efectuar o registo nesta tabela 3.16.

Interfaces						
Index(I)	**Nome(S)**	**OIDLocalizacao(S)**	**VC(R)**	**VL(R)**	**VA(R)**	**Ip/Porta(S)**
1	Comando DOMinho	OID.3.5:192.168.1.1:161	290	180	150	192.168.10.1:162
2	Comando DOMinho 2	OID.4.5:192.168.1.1:161	180	0	0	192.168.10.2:162

Tabela 3.16: Tabela de interfaces.

Esta tabela possui os seguintes atributos:

- index: identificador único de cada interface.
- nome: identifica o nome do interface.
- OIDLocalizacao: identifica qual a posição da interface no sistema. Este valor é calculado pelo software de inferências com base nas coordenadas geográficas da interface. Desta forma, sempre que esta localização é alterada, é enviada uma notificação à interface, de forma a ser gerado um novo menu de actividades pela camada de inferências.
- Vc, Vi, Vl: Permite a inserção de uma coordenada tridimensional na tabela, identificadora da posição exacta da interface.
- ip/porta: estes parâmetros são definidos pela interface, e identificam qual o ip e porta onde são esperadas notificações SNMP. Estes dois parâmetros são separados pelo caracter ":".

3.9.2 Tabela de características interfaces

A tabela 3.17 permite a identificação das características de uma dada interface, como por exemplo, o tipo ou os tipos de interacções que disponibilizam, a resolução e número de cores do seu ecrã, etc.

CaracterísticasInterfaces			
Index(I)	**IdxInterface(I)**	**IdxCaracterística(I)**	**OIDValor(VariablePointer)**
1	1	1	OID

Tabela 3.17: Tabela de características interface.

Os atributos definidos para esta tabela são:

- index: identificador único de cada característica.
- idxInterface: permite identificar a interface.
- idxCaracteristica: permite identificar a característica da interface correspondente à entrada da tabela. Todas as características disponíveis são definidas pela comunidade de domótica.
- OIDValor: permite a identificar o valor que é pretendido atribuir à característica. Caso exista mais do que um valor para a mesma característica, deve ser utilizado o carácter ":" para fazer a separação desses valores.

3.9.3 Tabela funcionalidades interfaces

A tabela 3.18 permite a identificação das funcionalidades de uma dada interface, como por exemplo, a possibilidade de alterar o tipo de letra, o tamanho de letra, o volume, etc. Estas funcionalidades expressam assim, a complexibilidade da interface, e definem quais as opções que os softwares podem utilizar para adaptarem os layouts e interacções de saída ao utilizador.

Os atributos definidos nesta tabela são:

- index: identificador único de cada entrada.

FuncionalidadeInterface		
Index(I)	**idxInterface(I)**	**IdxFuncionalidade(I)**
1	1	1
2	2	2

Tabela 3.18: Tabela de funcionalidades interface.

- idxInterface: permite identificar a interface.
- idxFuncionalidade: identifica a funcionalidade que é disponibilizada pela interface. Todas as funcionalidades são definidas pela comunidade de domótica.

3.9.4 Tabela interface utilizador

O objectivo desta tabela é indicar ao sistema que determinado utilizador está a utilizar uma determinada interface. Deste modo, sempre que é inserida uma nova entrada nesta tabela, é enviada uma notificação ao software de inferências identificando a necessidade da geração de um menu de actividades. Esta notificação contém as informações necessárias para a geração dos menus, o índice do utilizador e da interface.

InterfaceUtilizador		
Index(I)	**IdxInterface(1)**	**OIDUtilizador(S)**
1	1	OID.1:ip:porta
2	2	OID.2:ip:porta

Tabela 3.19: Tabela de interface utilizador.

Os atributos definidos para esta tabela são:

- index: identificador único de cada entrada.
- idxInterface: identifica a interface para a qual é necessário gerar o menu.
- idxUtilizador: identifica o utilizador, cujo software inteligente se deve basear para gerar o menu.

3.9.5 Tabela menu

Com esta tabela indica-se qual o tipo de menu que deve ser apresentado ao utilizador. Sempre que uma nova entrada é inserida nesta tabela, é enviada uma notificação para a interface correspondente com informações sobre o índice desta tabela e o tipo de menu a apresentar. A existência desta tabela permite também a apresentação de mais do que um menu numa interface no mesmo instante de tempo.

Menu		
Index(I)	**idxInterface(I)**	**IdxTipoMenu(I)**
1	1	1
2	2	2

Tabela 3.20: Tabela menu.

Os atributos modulados para esta tabela foram:

- index: identificador único de cada entrada.
- idxInterface: identifica a interface para a qual o menu foi gerado.
- idxTipoMenu: identifica o tipo de menu associado ao menu definido.

3.9.6 Tabela tipo de interacção do menu

A tabela 3.21 permite identificar qual o tipo de interacção que deve ser utilizado para apresentar um determinado menu. Torna-se assim possível, a utilização de mais do que um tipo de interacção para apresentar um menu ao utilizador. A utilização desta tabela é apenas necessária caso uma interface possua mais do que um tipo de interacção para comunicar com os utilizadores. De referir que, por cada entrada nesta tabela, é gerada uma notificação para a interface visada, de forma a indicar qual o meio, ou meios, de interacção que deve utilizar para comunicar com o utilizador.

- index: identificador único de cada entrada.
- idxMenu: identifica o menu para o qual é pretendida a interacção.

MenutipoInteracao		
Index(I)	**idxMenu(I)**	**IdxTipoInteracao(I)**
1	1	1
2	2	2

Tabela 3.21: Tabela do tipo de interacção do menu.

- idxTipoInteracao: identifica o tipo de interacção pretendida. Todos os tipos de interacção permitidos, encontram-se definidos na tabela da comunidade.

3.9.7 Tabela características de menu

Esta tabela foi modulada de forma a que os módulos de inferência possam determinar a forma como os menus devem ser apresentados aos utilizadores. Assim, características como a cor, o volume, o fundo, o tipo de letra, o tamanho de letra, o tamanho dos ícones, são passiveis de alterações por parte dos softwares inteligentes. Também nesta tabela, estas informações são enviadas à interface através do envio de notificações.

CaracteristicasMenu			
Index(I)	**IdxMenutipoInteracao(I)**	**IdxCaracteristica(S)**	**OIDValor(VariablePointer)**
1	1	1	OID

Tabela 3.22: Tabela características de menu.

Os atributos modulados foram:

- index: identificador único de cada entrada.
- IdxMenutipoInteracao: identifica o menu e o tipo de interacção para os quais a característica foi definida.
- idxCaracteristica: identifica a característica definida.
- OIDValor: atribui o valor pretendido para a característica definida.

3.9.8 Tabela actividade a apresentar ao utilizador

É a principal forma do sistema comunicar com o utilizador, sendo definidas todas as actividades que devem ser apresentadas num determinado menu de actividades. Cada nova entrada nesta tabela origina o envio de uma notificação para a interface, devendo enviar o campo status da tabela com o valor "0" quando se pretender indicar à interface que o menu não possui mais nenhuma actividade para uma dada interacção. Esta notificação contem informações sobre o índice da entrada, da tabela de menus, da tabela de actividades, o OID do equipamento e localização, o valor actual da actividade, a preferência do utilizador pela actividade e o status da entrada.

Por sua vez, quando o utilizador selecciona uma determinada actividade, a interface insere o OID do valor no campo OIDValor da entrada correspondente à actividade seleccionada, sendo gerada uma notificação para o software inteligente, com a informação do índice da entrada que possui a actividade seleccionada e o valor inserido pelo utilizador. Caso o valor inserido na tabela seja um "c", ou seja, o utilizador seleccionou uma actividade que possui actividades "descendentes", o sistema inteligente tem de gerar novo menu. Neste caso, o index do menu será o mesmo, sendo apenas inseridas as novas actividades inferidas na tabela.

ActividadesApresentar							
Index(I)	**IdxMenu(I)**	**IdxActividade(I)**	**OIDEquipamento(S)**	**OIDValor(VariablePointer)**	**ValorPreferencia(I)**	**OIDLocalização(S)**	**ValorActual(S)**
1	1	1	OID.1:ip:porta	OID	45	OID.2:ip:porta	Off
2	1	12	OID.3:ip:porta	OID	75	OID.2:ip:porta	21

Tabela 3.23: Tabela de actividade apresentar.

Esta tabela foi modulada com os seguintes atributos:

- index: identificador único de cada entrada.
- idxMenu: identifica o menu ao qual está associado a actividade.
- idxActividade: identifica a actividade que deve ser apresentada ao utilizador.
- OIDEquipamento: identifica o equipamento ao qual está associada a actividade. Através deste valor, é possível aceder a informações específicas do equipamento através do acesso à aplicação de serviços. Um exemplo para a utilização deste valor, pode ser dado para

a actividade “mudar canal”, onde a interface pode consultar quais os canais disponíveis neste equipamento. Caso não existam equipamentos associados a actividade, este atributo deve ser preenchido com o valor ’0’.

- OIDValor: Este campo permite que cada escolha do utilizador possa ser transmitida ao sistema de domótica. Sempre que este campo é preenchido, é enviada uma notificação para o software de inferências, gerado um log, e o seu valor é removido da tabela.

- valorPreferencia: identifica o valor de preferência do utilizador pela actividade inferida. Este valor pode ser utilizado pela interface para ordenar actividades por ordem preferencial do utilizador, ou para apresentar actividades de diferentes formas, tendo em consideração a preferência do utilizador por cada uma delas.

- OIDLocalização: identifica o local para onde a actividade foi gerada. Assim, caso o utilizador pretenda aceder a actividades disponíveis noutro local da casa que não o local onde se encontra, o software inteligente gera um novo menu, sendo identificado neste campo qual a localização pertencente à actividade gerada.

- valorActual: permite que seja identificado o valor actual de determinada actividade. Por exemplo, para a actividade 14 “Ver Temperatura”, este campo identifica o valor actual da temperatura. Sempre que este campo é alterado, é enviada uma notificação à interface, notificando-a da alteração do seu valor.

3.9.9 Tabela meta-informação

A tabela meta-informação permite complementar a tabela anterior, possibilitando que as preferências dos utilizadores sobre determinados temas possam ser reflectidas nas interfaces. Assim, preferências para a actividade “ver televisão”, como por exemplo, os canais favoritos e/ou os programas favoritos, podem ser indicados nesta tabela, assim como restrições atribuídas a estas duas preferências. Cada entrada inserida nesta tabela gera uma notificação para a interface. Estas notificações podem ser assíncronas, ou seja, as informações sobre meta-informações, o índice da entrada, do menu ao qual pertence a meta-informação, da actividade, do tipo de favorito como dos seus valores e restrições, podem ser enviadas à interface independentemente das actividades geradas.

MetaInformação					
Index(I)	**IdxMenu(I)**	**IdxActividade(I)**	**idxTipoFavorito(I)**	**Valores(S)**	**Restrições(S)**
1	1	1	1	RTP1:TVI	SportTv
2	1	1	3	Grande Entrevista	

Tabela 3.24: Tabela de meta informação.

- index: identificador único de cada entrada.
- idxMenu: identifica o menu para o qual foi criada a meta-informação.
- IdxActividade: identifica a actividade ao qual está associada esta meta-informação.
- idxTipoFavorito: identifica o tipo de favorito aa qual se refere a meta-informação. Todos os tipos de favorito são definidos pela comunidade de domótica.
- valores: permite a identificação dos valores preferidos do utilizador. Quando existe mais do que um valor associado a um favorito, estes devem ser separados pelo caracter ":".
- Restrições: permite a identificação restrições para o tipo de favorito inserido.

3.9.10 Tabela informação a apresentar ao utilizador

Esta tabela permite que todos os tipos de menus diferentes do menu "actividades", possam ser apresentados aos utilizadores. Tal como nas outras tabelas, cada nova entrada nesta tabela origina o envio de uma notificação para a interface.

InformaçãoApresentar			
Index(I)	**idxMenu(I)**	**OIDValor(VariablePointer)**	**informação(S)**
1	2		Maria, tens de tomar o medicamento

Tabela 3.25: Tabela de informação a apresentar.

Esta tabela possui assim os seguintes atributos:

- index: identificador único de cada entrada.

- idxMenu: permite identificar o menu associado à entrada.
- OIDvalor: este campo permite que o utilizador possa responder ao sistema. Deste modo, sempre que o utilizador responde à informação recebida, é despoletado uma notificação para o software de inferências com a sua resposta.
- Informação: permite a inserção da informação que se pretende apresentar ao utilizador.

3.9.11 Tabela de actividades solicitadas pelo utilizador

Através da tabela 3.26, o utilizador pode activar determinada actividade directamente no sistema. Esta tabela foi modulada para que interacções através da voz ou de gestos possam ser utilizadas para controlar serviços do sistema. Sempre que uma entrada é inserida nesta tabela, é enviada uma notificação ao software de inferências.

ActividadesSolicitadasUtilizador					
Index(I)	**idsUtilizador**	**idxInterface**	**IdxActividade(I)**	**OIDEquipamento(S)**	**OIDValor(VariablePointer)**
1	2	2	1	OID.2:ip:porta	OID

Tabela 3.26: Tabela de actividades solicitadas pelo utilizador.

Deste modo esta tabela foi modulada com os seguintes atributos:

- index: identificador único de cada entrada.
- idxUtilizador: identifica o utilizador.
- idxInterface: identifica a interface utilizada pelo utilizador.
- IdxActividade: identifica a actividade que o utilizador pretende controlar.
- OIDEquipamento: identifica o equipamento que é pretendido controlar.
- OIDValor: permite atribuir o valor atribuído pelo utilizador à actividade.

3.9.12 Tabela características de actividades a apresentar

Por fim, esta tabela complementa a tabela "Características Menu". Assim, além de se permitir que o software de inferências possa definir a forma como determinado menu é apresentado, é também possível definir a forma como uma actividade específica deve ser apresentada ao utilizador. Tal como na tabela de "características menu" também aqui é gerada uma notificação para a interface.

ActividadesAapresentar_Caracteristicas				
Index(I)	**IdxActividadesAapresentar(I)**	**idxInterfacetipoInteracao(I)**	**IdxCaracteristica(S)**	**OIDValor(VariablePointer)**
1	1	1	1	OID

Tabela 3.27: Tabela de características de actividades a apresentar.

Esta tabela possui os seguintes atributos:

- index: identificador único de cada entrada.
- idxInterfaceAApresentar: identifica o índice da tabela ao qual pertence a característica definida.
- idxInterfaceTipoInteracao: identifica o tipo de interacção e o menu para a qual esta característica se pretende reflectir.
- IdxCaracteristica: identifica a característica que se pretende definir.
- OIDValor: permite atribuir o valor pretendido para a característica.

Capítulo 4

Protótipo

Tal como referido, os sistemas desenvolvidos utilizam o protocolo de comunicação SNMP. Para tal, foram criadas duas MIBs, apresentadas em anexo, que permitem a definição de todas as tabelas moduladas. Estas MIBs, necessitam de ser registadas num agente SNMP de forma a que os gestores SNMP possam aceder às suas informações.

Para a criação dos gestores SNMP necessários, foi utilizada a API *open source* SNMP4j [73]. Esta API, disponibilizada para JAVA, além de fornecer o suporte de todas as versões SNMP apresenta as seguintes características:

- Autenticação MD5 e SHA e privacidade DES, AES 128, AES 192 e AES 256 na terceira versão do SNMP;
- Permite a utilização de todos os tipos de PDU;
- Suporte de protocolos UDP e TCP;
- Pedidos síncronos e assíncronos;
- Suporte multi-thread;

Para a geração do código JAVA dos agentes SNMP, recorreu-se ao software AgenProv3 [74] que cria toda a estrutura de Objectos definida na MIB, bem como os métodos de criação, acesso e armazenamento das entradas das tabelas definidas na MIB. Além da criação do código, permite também a compilação de MIBs e a geração de código JAVA ou C++.

Este código, foi alterado de forma a possibilitar a execução de determinadas acções. Cada um dos módulos, possuem assim um agente SNMP onde são registadas as MIB criadas e um gestor SNMP para acesso às tabelas disponíveis nos restantes agentes do sistema DOMinho. Para que os agentes sejam capazes de enviar as notificações especificadas, foi necessário definir quais as situações que as devem originar. Para tal, foi necessário adicionar um "listener" às tabelas disponibilizadas pelos agentes. Deste modo, sempre que determinado valor é alterado, este listener executa um método chamado "rowChanged". Através deste método, é possível verificar o campo e valor alterado e executar o envio da notificação definida na MIB, caso pretendido.

Para que o sistema seja implementado com a generalidade pretendida, foram criados ficheiros de configuração que permitem que novas informações possam ser adicionadas, sem a necessidade da alteração de código. Cada módulo disponibiliza ainda um menu onde podem ser impressas todas as entradas das suas tabelas. Este menu permite a validação do funcionamento do sistema de forma rápida e simples.

4.1 Módulo de utilizadores

De forma a que as funcionalidades propostas sejam disponibilizadas, foi acedido ao código gerado pela aplicação AgenProv3 e acrescentados métodos que de forma rápida e simples suportam os requisitos propostos.

Assim, quando o módulo é arrancado, é lido um ficheiro denominado de "ValoresEsteriootipos" e guardados os seus conteúdos na tabela de estereótipos. Estes valores são utilizados posteriormente no registo de novos utilizadores no sistema. Nesse momento, é verificado o estereótipo no qual o utilizador se enquadra, tendo em consideração as informações pessoais facultados pelo próprio no acto do seu registo. Após esta verificação, as informações definidas nesse estereótipo são copiadas para a tabela perfil de utilizador.

Como referido, é pretendido conhecer dinamicamente os hábitos e comportamentos dos utilizadores. Para tal, é realizado um cálculo do nível de confiança atribuído a cada interacção realizada pelo utilizador. Este cálculo, é realizado periodicamente, com base nas informações disponibilizadas por um ficheiro denominado de "logsUtilizador" concatenado com o identificador do utilizador, que contem a informação de todas as interacções realizadas pelo utilizador no sistema de domótica. Para tal, foi criada uma "thread" com uma periodicidade configurável,

que calcula e armazena o nível de confiança associado a cada actividade.

Esta "thread", efectua a leitura dos ficheiros disponibilizados, que possuem uma nomenclatura sequencial, ou seja, cada nova entrada é escrita numa nova linha. Todas as informações são separadas pelos caracteres "**", e disponibilizam as seguintes informações:

- idxActividade: identificador da actividade seleccionada pelo utilizador.
- data/hora início Actividade: data em que a actividade foi seleccionada.
- data/hora fim Actividade: permite identificar o tempo em que determinada actividade se mantém com o mesmo valor. Apesar deste parâmetro poder não ser relevante para todas as entradas desta tabela, pode ser utilizado por exemplo, para uma situação onde o utilizador efectua "zapping" pelos canais de televisão disponíveis. Neste caso, através da utilização deste valor, os cálculos do nível de confiança de cada canal, têm uma maior fiabilidade, podendo também neste caso, não ser considerados canais que possuem menos de um determinado tempo de visualização.
- valor: este campo determina o valor seleccionado pelo utilizador para determinada actividade. Caso a actividade seja de alto nível, como por exemplo "ver televisão", este campo é preenchido com o carácter "c".
- contexto de utilizador: neste campo são definidos os identificadores de todos os utilizadores presentes no mesmo espaço onde a actividade foi seleccionada. Caso este campo possua mais do que um valor, é utilizado o carácter ":" para realizar a separação dos identificadores de cada utilizador.
- OID:ip:porta localização: este parâmetro permite a identificação do espaço onde a actividade foi seleccionada.
- OID:ip:porta equipamento: este campo permite a identificação do equipamento onde a actividade foi despoletada.
- index Interface: permite a identificação do interface pelo qual o utilizador interagiu com o serviço.

A partir das informações disponibilizadas por este ficheiro, o módulo de utilizadores realiza o processo apresentado em 4.1.

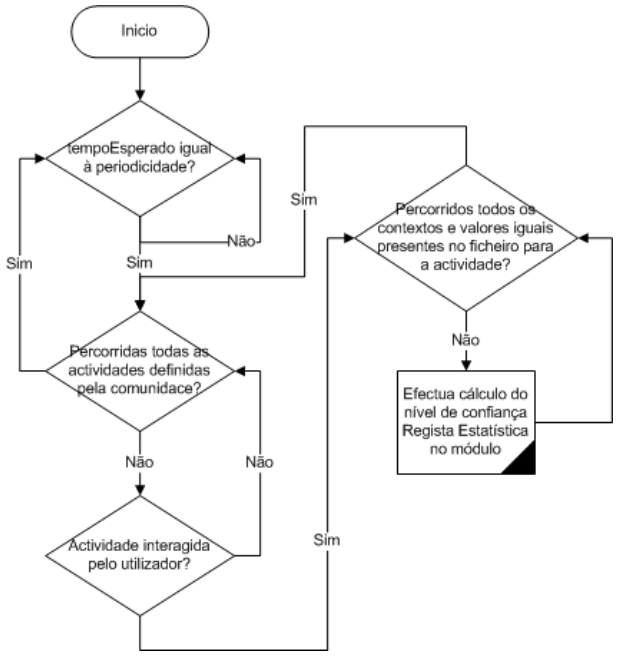


Figura 4.1: Fluxograma do cálculo de estatísticas.

O cálculo do valor do nível de confiança é realizado através da utilização da fórmula:

$$nivelConfianca = \frac{numeroDeActividadesIguaisComContextosEvalorIguais}{numeroTotalDeActividadesSelecionadas} * 100 \tag{4.1}$$

De modo a que a aplicação de utilizadores possa ser o mais dinâmica possível, foram criados ficheiros de configuração para que novas informações da comunidade de domótica e de rede possam ser facilmente alteradas e adicionadas. Os ficheiros criados foram:

- configurações: neste ficheiro pode ser definido o ip e porta do agente snmp, bem como o ip e porta do servidor de notificações do software inteligente. É também possível definir se é pretendido guardar as informações em ficheiro, e indicar o nome desse ficheiro. Por fim, é também possível definir qual a periodicidade (segundos) da frequência com que é pretendido gerar estatísticas e o caminho para o directório onde podem ser encontrados os ficheiros de logs.

- valoresEsteriotipos: Neste ficheiro são definidos todos os valores definidos para cada estereótipo definido pela comunidade de domótica. Cada linha deste ficheiro corresponde

a uma entrada na tabela de estereótipos, e tem definido o index da tabela, o index da tabela de estereótipos ao qual se refere o estereótipo da comunidade, o index da tabela perfil de utilizador, o valor atribuído ao estereótipo/perfil e o nível de confiança que é dado ao valor definido.

- Actividades: O ficheiro de actividades permite que o sistema conheça todas as actividades existentes no sistema de forma a serem realizadas estatísticas. Como já foi referido as actividades são o conjunto de opções que os utilizadores têm ao seu dispor para controlar o sistema.

- Autenticação: este ficheiro permite que as credenciais de cada utilizador sejam guardadas.

4.2 Módulo de interfaces

Este componente permite que todas as interfaces possam interagir com o sistema DOMinho. Além da disponibilização das informações presentes nas suas tabelas, este módulo efectua o envio das notificações definidas na MIB. Para tal, também aqui o código fonte criado pelo AgenProv3 foi alterado, de forma a possibilitar a execução de determinadas acções, quando certos atributos são alterados.

Um exemplo deste caso, é a disponibilização do ficheiro com a informação das interacções dos utilizadores. Neste caso, cada entrada inserida neste ficheiro é gerada quando o utilizador executa uma determinada acção no sistema. Para tal, é realizado o processamento apresentado no fluxograma 4.2.

De modo a que este módulo seja capaz de albergar facilmente novas informações, foram criados um conjunto de ficheiros que permitem uma rápida configuração e actualização dos seus valores. Os ficheiros criados foram:

- configurações: tal como para o sistema de utilizadores, neste ficheiro pode ser definido o ip e porta do agente snmp, o ip e porta do servidor de notificações do software inteligente. É também possível definir se é pretendido guardar as informações em ficheiro, e indicar o nome desse ficheiro. Por fim, é definido qual o caminho para o directório onde são guardados os ficheiros de logs criados.

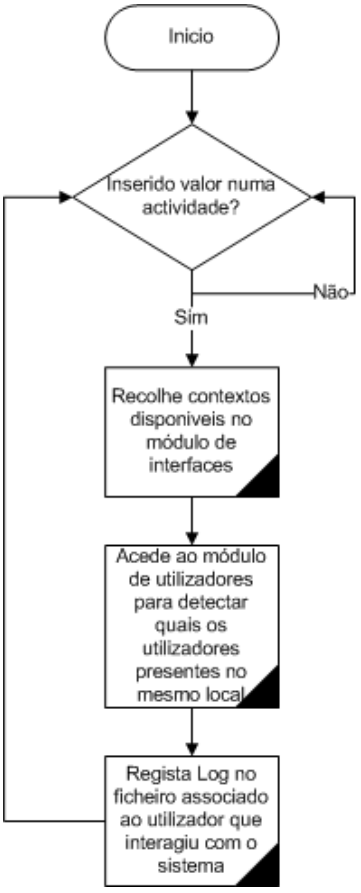


Figura 4.2: Fluxograma do registo das interacções dos utilizadores.

- OIDsUsers: possui a relação entre os nomes das tabelas de utilizadores e os seus OIDs. Com esta informação, sempre que o agente de interfaces necessite comunicar com o de utilizadores, não necessita de conhecer o seu OID, bastando conhecer o nome da tabela à qual pretende aceder. Apesar de todos os OIDs serem estáticos, este ficheiro foi criado para a fase de desenvolvimento, onde possíveis alterações nas tabelas poderiam acontecer.

- SNMPConfUsers: neste ficheiro são definidas as configurações que o sistema necessita de conhecer para comunicar com o agente de utilizadores. Desta forma, a versão do protocolo SNMP, a password, o protocolo, o ip e porta, o intervalo de amostragem e o tempo de timeout são definidos neste ficheiro.

- ficheiro de logs: tal como referido, este agente cria um ficheiro de logs por utilizador, de forma a poder existir um conhecimento dinâmico dos hábitos e comportamentos de cada utilizador.

4.3 Protótipo interface

Este protótipo foi desenvolvido com recursos à tecnologia swing do java, e tal como nos componentes de utilizadores e interfaces, recorreu-se à API SNMP4j [73], para a criação de um gestor SNMP para acesso aos módulos desenvolvidos e para a recepção de notificações SNMP.

O protótipo desenvolvido disponibiliza assim uma interface gráfica que adapta os seus conteúdos com base na informação recebida do sistema de interfaces. Além das primitivas SNMP para acesso e modificação de conteúdos, possui também um servidor de notificações, pelo qual recebe as informações dos menus que deve apresentar aos utilizadores. Possibilita ainda a adaptação à linguagem do utilizador sem que seja necessária a intervenção da camada de inferências.

De forma a que todos os dados possam ser rapidamente alterados e adicionados na interface, foram criados 16 ficheiros de configuração, os quais permitem configurar e conhecer as informações definidas pela comunidade de domótica:

- SNMPConfNotifi: neste ficheiro é definido as configurações relativas ao servidor de notificações SNMP. São aqui definidos a versão do protocolo SNMP, o ip, a porta e tempo de timeout para as notificações.
- SNMPConfUsers: neste ficheiro são definidas as configurações necessárias para a comunicação com o agente de utilizadores, são assim definidos a versão do protocolo SNMP, a password, o tipo de protocolo, o ip, a porta, o intervalo de amostragem e o tempo de timeout.
- SNMPConfInterfaces: da mesma forma, neste ficheiro são definidos as configurações necessárias para efectuar a comunicação com o agente de interfaces.
- SNMPConfServicos: mais uma vez, este ficheiro permite definir as configurações necessárias para efectuar a comunicação com o agente de serviços.
- OIDsUsers: possui as relações entre os nomes das tabelas de utilizadores e os seus OIDs.
- OIDsInterfaces: possui as relações entre os nomes das tabelas de interfaces e os seus OIDs.
- OIDsServicos: possui as relações entre os nomes das tabelas de serviços e os seus OIDs.

- Actividades: O ficheiro de actividades permite identificar quais as actividades disponíveis no sistema DOMinho. Através deste ficheiro, é possível correlacionar o identificador dos indexes que são inseridos nas tabelas com os nomes das actividades. Neste ficheiro, é também atribuído um caminho para uma imagem, que é utilizada para representar a actividade, e é dada uma indicação do tipo de actividade, ou seja, é indicado se a actividade possui ou não actividades descendentes. Com estes dados, é também possível identificar qual o código de uma determinada actividade, caso o interface possua algum tipo de mecanismo de interacção inteligente e seja feita uma interacção directa pelo utilizador no sistema.

- ActividadesInglês: este ficheiro é uma cópia do ficheiro anterior, mas aqui os nomes das actividades são definidas em Inglês.

- PerfilUtilizadorComunidade: este ficheiro permite identificar todos os perfis de utilizador definidos pela comunidade de domótica. É assim representado o código do perfil, o nome do perfil, o código do perfil pai, o tipo de valor que pode tomar o perfil. Em caso do perfil poder tomar uma lista sequencial, cada um dos valores é inserido após o código do tipo de valor possível. Para o caso do valor poder tomar um valor tuplo, é inserido o valor mínimo, máximo e o valor incremental que é possível dar entre valores.

- PerfilUtilizadorComunidadeInglês: este ficheiro é uma cópia do ficheiro anterior, mas aqui os nomes das actividades são definidos em Inglês.

- TipoInteracao: possui a relação entre os identificadores do tipo de interacção, e o nome do tipo de interacção.

- TipoMenu: possui a relação entre os identificadores do tipo de menus, e o nome do menu.

- TipoFavorito: possui a relação entre o código do tipo de favorito e a nome do favorito.

- TipoUtilizador: possui a relação entre o código do tipo de utilizador e a nome do tipo de utilizador.

- Caracteristicas: possui a relação entre o código das características definidas, e o nome das mesmas. É também referido qual a unidade de cada característica.

4.3.1 Arranque interface

Sempre que a interface desenvolvida é activa, este efectua um processo de registo e/ou actualização no sistema. Neste processo são verificadas as ligações com o agente de utilizadores e interfaces e serviços. Após este teste, é lançado o servidor de notificações. Estes passos permitem que o funcionamento da interface só seja possível caso consiga comunicar com os módulos dos quais esta dependente e permite que todos os seus dados sejam conhecidas pelo sistema, de forma a que as adaptações de interfaces possam ser desde logo possíveis. Por outro lado, através da criação do servidor de notificações permite desde do seu arranque que o sistema possa interagir com o utilizador. O fluxograma 4.3 apresenta este processo.

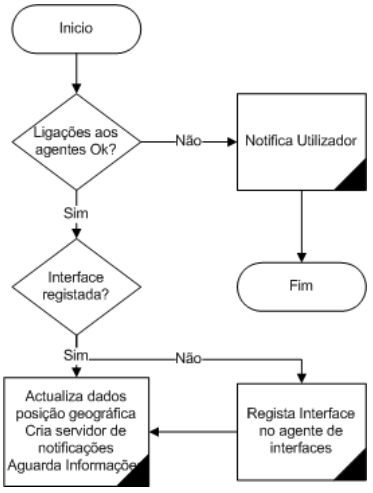


Figura 4.3: Fluxograma do arranque da interface.

4.3.2 Gestão de notificações

Como é possível verificar ao longo da exposição das tabelas, existem várias notificações que podem ser recebidas pelas interfaces. Estas notificações, permitem retirar um peso considerável do processamento, e da rede, caso fosse considerada uma arquitectura baseada em base de dados, pois neste caso seria necessário efectuar um pooling a todas as tabelas.

Para que as notificações possam ser processadas pela interface, foi criada uma "thread" que as recebe e guarda-as numa queue do tipo fifo, para que possa ficar à escuta de novas notificações. O fluxograma 4.4 apresenta o funcionamento deste processo.

Figura 4.4: Fluxograma da thread de notificações.

Sempre que verificado a existência de novos elementos na queue de notificações, é realizado um processo de verificação do tipo de notificação recebida, utilizando para isso o OID da notificação. Depois dessa detecção, os seus valores são formatados de forma mais simples e colocados na queue respeitante ao seu tipo. Posteriormente esta notificação é processada e a interface apresentada ao utilizador. O fluxograma 4.5 apresenta o funcionamento deste processo.

Figura 4.5: Fluxograma da verificação do tipo de notificação recebida.

4.4 Resultados experimentais

Uma vez terminado a apresentação do protótipo proposto, torna-se necessário testar o trabalho desenvolvido. Este teste visa demonstrar a viabilidade dos conceitos apresentados e fomentar a discussão sobre as vantagens da abordagem seguida.

Deste modo, foi criado um cenário de testes no âmbito do projecto DOMinho, sendo utilizadas as quatro camadas descritas em 1.1. Para tal, foram utilizados quatro computadores ligados numa rede Ethernet, onde toda a comunicação entre as diferentes camadas é feita única e exclusivamente através do protocolo SNMP, como definido pelo projecto DOMinho. Devido à inexistência de um módulo UWB, o protótipo da interface criada foi testada juntamente com o sistema de utilizadores e interfaces desenvolvido.

O cenário desenhado consiste numa habitação térrea com cinco divisões, em que cada divisão possui diferentes equipamentos. Estes equipamentos são representativos de dois serviços domóticos, climatização e multimédia. O serviço de multimédia possui apenas duas televisões, que permitem ao utilizador a alteração de canal e a regulação de volume. Por sua vez o serviço de climatização permite regular, programar e verificar a temperatura que das diferentes divisões da habitação.

Com a utilização deste cenário é pretendido demonstrar:

- o comportamento da interface perante uma configuração de rede incorrecta.
- o arranque da interface com configurações correctas.
- o registo de um novo utilizador no sistema.
- um menu do tipo aviso.
- a apresentação do menu principal.
- o acesso a uma divisão da casa, a partir de outro divisão
- a programação da temperatura de um divisão a partir de outro.
- a simulação da alteração de divisão da habitação.
- o acesso à actividade “Ver Televisão”.
- as estatísticas geradas pelo sistema de utilizador.

Após a apresentação do cenário proposto, torna-se necessário activar a aplicação de utilizadores e interfaces.

No arranque do sistema de utilizadores, tal como esperado, é carregada a tabela de estereótipos com base nas informações do ficheiro “ValoresEsteriotipos”. Na figura 4.6 são apresentadas

duas das 140 entradas inseridas pela aplicação de utilizadores nesta tabela.

```
Numero de entradas na Tabela Estereótipos = 140

Indice = 1
Idx Esteriotipo = 2
Idx perfil de utilizador = 15
OID value = 1.3.6.1.3.79.9.2.1.2.1
Valor = 88
Nivel de confianca = 45

Indice = 2
Idx Esteriotipo = 2
Idx perfil de utilizador = 16
OID value = 1.3.6.1.3.79.9.2.1.2.2
Valor = 53
Nivel de confianca = 84
```

Figura 4.6: Estereótipos.

As duas entradas apresentadas são correspondentes ao estereótipo 2 (homem). Na primeira entrada foi atribuído ao perfil de utilizador 15 (preferências/desporto/**baseball**), uma preferência de 88% com um nível de confiança de 45%. Já para a segunda entrada é atribuído ao perfil 16 (preferências/desporto/**basketball**), o valor 53 com um nível de confiança de 84 Com esta associação de valores, é possível atribuir um valor médio às preferências do utilizador desde o seu registo.

4.4.1 Comportamento da interface perante uma configuração incorrecta

Depois de iniciadas as aplicações, é possível activar a interface e comunicar com todo o sistema DOMinho. Neste caso, a interface arranca com a configuração do IP e porta do módulo de serviços incorrectas. Pretende assim validar-se se a interface notifica o utilizador do sucedido. A interface apresentada ao utilizador nesta situação é apresentada na figura 4.7.

Como é possível verificar através da interface, é apresentada uma notificação ao utilizador com a indicação do erro que impossibilita o seu correcto funcionamento.

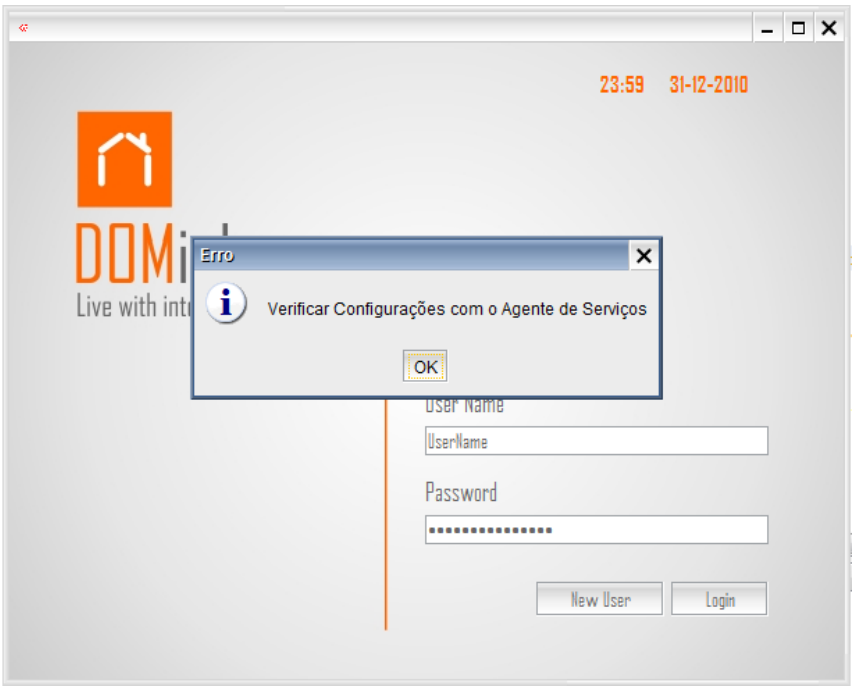


Figura 4.7: Interface arranque com configurações incorrectas.

4.4.2 Arranque de interface com configurações correctas

Voltando a correr a interface, mas desta vez com as configurações correctas, as tabelas de interfaces 4.8 e de características de interfaces 4.9, são preenchidas. A tabela de funcionalidades não é preenchida, pois, tal como foi referido, a interface não possui nenhum tipo de funcionalidade que permita alterar o seu layout de saída.

Na figura 4.8, podem ser visualizadas as informações preenchidas pela interface na tabela de interfaces. A localização, é preenchida pela camada de inferências com base nas coordenadas geográficas inseridas.

Na figura 4.9 são apresentadas as entradas correspondentes às características da interface. Deste modo, pode ser confirmado que a característica 8 (tamanho visor), possui o valor "1366:768" indicado no OID 1.3.6.1.3.80.15.2.1.2.1. A segunda característica possui o identificador 6 (tipo de interacção), cujo valor encontra-se no OID 1.3.6.1.3.80.13.2.1.2.1, correspondente ao valor "Gráfica".

Após o registo da interface no sistema é apresentado o menu de login 4.10. Pretende-se, no entanto, que este menu não seja apresentado no futuro, por se desejar que todas as autenticações

```
Numero de entradas na tabela Interfaces = 1

Indice = 1
Nome Interface = Interface DOMinho
OIDLocalizacao = 1.3.6.1.3.78.2.2.1.3.2:192.168.2.5:1661
Vc = 3
Vl = 3
Va = 3
IP e Porta de notificações = 127.0.0.1:162
```

Figura 4.8: Tabela de interfaces.

```
Numero de linhas da tabela caracteristicas de interfaces = 2

Indice = 1
Idx Interface = 1
Idx Caracteristica = 8
Value OID = 1.3.6.1.3.80.15.2.1.2.1
Valor = 1366:768

Indice = 2
Idx Interface = 1
Idx Caracteristica = 6
Value OID = 1.3.6.1.3.80.15.2.1.2.2
Valor = Grafica
```

Figura 4.9: Características interfaces.

sejam realizadas através de leituras biométricas.

4.4.3 Registo do utilizador no sistema

Após o arranque da interface, o utilizador pode interagir com todo o sistema. Para tal necessita, numa primeira fase, de se registar no sistema: deve inserir o seu username e password. A interface interage assim com a tabela de autenticação e retorna o identificador do utilizador no sistema. Devido à rapidez do processo torna-se impossível apresentar a tabela de autenticação preenchida, uma vez que a entrada desta tabela é apagada automaticamente pela aplicação de interfaces.

Figura 4.10: Interface login.

Após este registo são apresentados três menus que recolhem informações pessoais do utilizador. As informações solicitadas ao utilizador, seguem os perfis e seus valores possíveis, definidos pela comunidade de domótica. Ver figura 4.11.

Figura 4.11: Interface registo de utilizador.

A segunda interface, apresentada na figura 4.12 possui a particularidade de adaptar o seu idioma ao utilizador. Esta adaptação depende da nacionalidade e idiomas inseridos.

Figura 4.12: Interface registo de utilizador 2.

Como pode ser comprovado, todas as interfaces até ao momento têm sido apresentadas em Inglês. No entanto, na figura 4.13 o idioma foi alterado para Português.

Por fim, na figura 4.14 é apresentada a última interface de recolha de dados pessoais do utilizador. Após o utilizador completar o registo são preenchidas as tabelas de "utilizador", "perfil de utilizador" e de "interfaces utilizador".

Na tabela de utilizadores, apresentada em 4.15, é preenchido o username do utilizador, o tipo de utilizador, tendo neste caso, o valor "1" referente ao tipo administrador. A localização e os vectores das coordenadas geográficas possuem o valor "0", dada a inexistência de um localizador de utilizadores.

Na tabela de perfis de utilizador são preenchidos todos os detalhes pessoais recolhidos, assim como os perfis médios preenchidos pela aplicação de utilizadores baseado no estereótipo em

Figura 4.13: Interface registo de utilizador 3.

Figura 4.14: Interface registo de utilizador 4.

```
Numero de entrada na tabela Utilizadores = 1

Indice = 1
UserName = Miguel Marques
Idx Tipo de utilizador = 1
OID Localizacao = 0
Vc = 0
Vl = 0
Va = 0
Status = 1
```

Figura 4.15: Utilizadores.

que o utilizador é mapeado. Na figura 4.16 são apresentados dois detalhes pessoais e dois perfis preenchidos pelo sistema nesta tabela.

Como é visível na figura, foi inserido para o utilizador 1, o perfil de utilizador 3 (sexo do utilizador), o qual possui o OID 1.3.6.1.3.79.11.2.1.2.3, correspondente ao valor "Male". Já o perfil de utilizador 4 (data de nascimento) possui o valor "1986-9-23" cujo OID é 1.3.6.1.3.79.11.2.1.2.4. Para os perfis referidos não existe nenhum contexto associado e o nível de confiança atribuído a este valor é de 100%, pois é um detalhe pessoal inserido pelo utilizador.

Por sua vez, as entradas 10 e 11 desta tabela possuem dois dos valores preenchidos pela aplicação de utilizadores baseados na tabela de estereótipos. Neste caso, o utilizador foi mapeado no estereótipo "Adulto". Deste modo, os valores destas duas entradas correspondem aos valores apresentados na tabela de estereótipos 4.6.

Após o preenchimento de todos os dados relativos ao utilizador, a interface preenche a tabela "interface utilizador" apresentada na figura 4.17. Esta tabela tem como função indicar ao sistema de inferências que deve gerar um menu de actividades para o utilizador 1 que está a utilizar a interface 1.

Após o preenchimento desta tabela, a interface fica a aguardar que lhe sejam enviadas notificações com informações que permitam construir a interface a apresentar ao utilizador.

```
Numero de entradas na tabela perfis de utilizador = 44

Indice = 3
Idx utilizador = 1
Idx perfil de utilizador = 6
OID = 1.3.6.1.3.79.11.2.1.2.3
Valor = Male
nivel confianca = 100
IdxContexto = 0

Indice = 4
Idx utilizador = 1
Idx perfil de utilizador = 7
OID = 1.3.6.1.3.79.11.2.1.2.4
Valor = 1986-9-23
nivel confianca = 100
IdxContexto = 0


Indice = 10
Idx utilizador = 1
Idx perfil de utilizador = 15
OID = 1.3.6.1.3.79.9.2.1.2.141
Valor = 88
nivel confianca = 45
IdxContexto = 0

Indice = 11
Idx utilizador = 1
Idx perfil de utilizador = 16
OID = 1.3.6.1.3.79.9.2.1.2.142
Valor = 53
nivel confianca = 84
IdxContexto = 0
```

Figura 4.16: Perfis de utilizador.

```
Numero de entradas na tabela interface utilizador = 1

Indice = 1
idx Interface = 1
OID Utilizador = 1.3.6.1.3.79.2.2.1.1.2:127.0.0.1:161
Status = 1
```

Figura 4.17: Interface utilizador.

4.4.4 Menu do tipo aviso

Após o registo do utilizador, e tal como referido no cenário proposto, é esperado a geração de um menu do tipo "Aviso". Este menu é apresentado ao utilizador após a interface receber duas notificações, uma a indicar a existência de um novo menu, apresentada na figura 4.18, e outra com a informação que é pretendido fornecer ao utilizador, apresentada na figura 4.19.

Na figura 4.18 são apresentados os menus gerados pelo software de inferências. Tal como esperado, foi gerado um menu do tipo 2 (aviso), no entanto, existem duas entradas nesta tabela. Isto deve-se ao facto da interface ter solicitado um menu de actividades, quando acabou de registar o utilizador no sistema.

```
Numero de entradas na tabela de menus = 2

Indice = 1
idx Interface = 1
idx Tipo de Menu = 1
Status Menu = 1

Indice = 2
idx Interface = 1
idx Tipo de Menu = 2
Status Menu = 1
```

Figura 4.18: Menu.

Após a recepção da notificação do tipo de menu gerado, a interface recebe a informação que é pretendida apresentar ao utilizador. Desta forma, o software de inferências insere uma entrada na tabela de "informaçõesAApresentar" apresentada em 4.19, que origina o envio de uma notificação para a interface.

```
Numero de entradas na tabela informações a apresentar ao utilizador = 1

Indice = 1
idx Menu = 2
OID Equipamento = 0
Informação = Miguel, a janela da cozinha encontra-se aberta
Status = 1
```

Figura 4.19: Informações a apresentar.

Esta tabela contém o identificador do menu correspondente à tabela de menus e a informação que deve ser apresentada ao utilizador. Neste caso, o valor do equipamento foi colocado a zero, pois a informação não é relativa a nenhum equipamento.

Após a recepção da informação destas duas tabelas a interface apresenta o menu 4.20. Tal como esperado, além do menu do tipo aviso, é também apresentado o menu de actividades, pois a interface já recebeu as notificações referentes ao menu de actividades.

Figura 4.20: Interface menu do tipo aviso.

Por fim, é de salientar, que neste cenário a interface não recebe nenhuma notificação com a identificação do tipo de interacção que deve utilizar. Isto deve-se ao facto da interface possuir apenas um tipo de interacção, como pode ser comprovado na tabela de características de interfaces apresentada na figura 4.9.

4.4.5 Interface principal

Como referido, foram gerados dois menus pelo software de inferências, o menu do tipo 2, já descrito, e o menu do tipo 1. Para a geração do último menu foi utilizada a tabela de actividades a apresentar, cujas entradas são apresentadas nas figuras 4.21 e 4.22.

```
Numero de entradas na tabela Actividades a Apresentar ao utilizador = 5

Indice = 1
Idx Menu = 1
Idx actividade = 1
OID Equipamento = 1.3.6.1.3.78.3.2.1.2.1:192.168.2.5:1661
OID Valor = null
Valor da preferencia para a actividade = 80
OID Localizacao = 1.3.6.1.3.78.2.2.1.3.2:192.168.2.5:1661
Valor Actual da actividade = Off
Status = 1

Indice = 2
Idx Menu = 1
Idx actividade = 5
OID Equipamento = 1.3.6.1.3.78.3.2.1.2.3:192.168.2.5:1661
OID Valor = null
Valor da preferencia para a actividade = 60
OID Localizacao = 1.3.6.1.3.78.2.2.1.3.2:192.168.2.5:1661
Valor Actual da actividade = null
Status = 1

Indice = 3
Idx Menu = 1
Idx actividade = 8
OID Equipamento = 0
OID Valor = null
Valor da preferencia para a actividade = 40
OID Localizacao = 1.3.6.1.3.78.2.2.1.3.2:192.168.2.5:1661
Valor Actual da actividade = null
Status = 1
```

Figura 4.21: Actividades a apresentar.

Para cada actividade inferida, são transmitidas à interface algumas informações que lhe dão o conhecimento dos seus detalhes. Por exemplo, para a actividade 1 (ver televisão), é definido o valor do OID do equipamento que permite a identificação das características, funcionalidades e localização do equipamento a partir do acesso à aplicação de serviços. O valor actual recebido permite identificar o estado do equipamento, ou seja, se este se encontra ligado ou desligado. Por último, a informação do valor da preferência permite atribuir o conhecimento da preferência do utilizador para a actividade em questão. Todas as restantes actividades recebidas pela interface,

```
Indice = 4
Idx Menu = 1
Idx actividade = 13
OID Equipamento = 0
OID Valor = null
Valor da preferencia para a actividade = 45
OID Localizacao = 1.3.6.1.3.78.2.2.1.3.2:192.168.2.5:1661
Valor Actual da actividade = null
Status = 1

Indice = 5
Idx Menu = 1
Idx actividade = 14
OID Equipamento = 1.3.6.1.3.78.3.2.1.2.3:192.168.2.5:1661
OID Valor = null
Valor da preferencia para a actividade = 50
OID Localizacao = 1.3.6.1.3.78.2.2.1.3.2:192.168.2.5:1661
Valor Actual da actividade = 21
Status = 0
```

Figura 4.22: Actividades a apresentar 2.

possuem as mesmas informações para actividades distintas. O resultado visível ao utilizador é a interface na figura 4.23.

Através da imagem apresentada é possível verificar que o nome do utilizador, a data/hora, e a localização da interface são apresentadas na interface. Também a actividade 14 (verificar temperatura) é imediatamente apresentada ao utilizador. É possível verificar ainda que as actividades inferidas são apresentadas ao utilizador por ordem preferencial. A opção "definições" é criada pela interface de modo a que o utilizador possa alterar o layout da interface, os perfis de utilizador, entre outros.

4.4.6 Acesso a outra divisão da casa a partir do divisão actual

Seguindo o cenário proposto, o utilizador realizará o acesso a outra divisão da habitação a partir do qual se encontra. Para tal, é seleccionada a actividade 13 (outras divisões). Esta selecção gera uma pesquisa na tabela de localizações da aplicação serviços, que visa a recolha de todas

Figura 4.23: Menu principal.

as localizações disponíveis no sistema de domótica. O resultado para o utilizador é apresentado na figura 4.24.

Figura 4.24: Menu outras divisões.

Após a selecção da divisão pretendida, neste caso a cozinha, é inserido o valor correspondente à sua localização no campo valor da actividade 13. Esta acção gera o envio de uma notificação para o software de inferências. Por sua vez, o software de inferências, envia um novo menu para a divisão seleccionada. Assim, são geradas notificações das novas actividades para a divisão seleccionada. Após a sua recepção é apresentada a interface 4.25 ao utilizador.

Figura 4.25: Interface cozinha.

4.4.7 Programação da temperatura

Através da interface 4.25 o utilizador tem ao seu dispor todas as actividades do divisão seleccionado. Tal como indicado no cenário, o utilizador selecciona a actividade regular temperatura.

Após a selecção desta actividade, a interface acede à aplicação de serviços e recolhe informações das temperaturas permitidas pelo equipamento. Estes valores são utilizados de forma a limitar os valores de temperatura seleccionáveis. Ver figura 4.26.

Através desta interface o utilizador selecciona a programação pretendida para a temperatura e regista-a. Esta acção provoca o preenchimento de uma entrada da tabela contexto temporal, apresentada em 4.27.

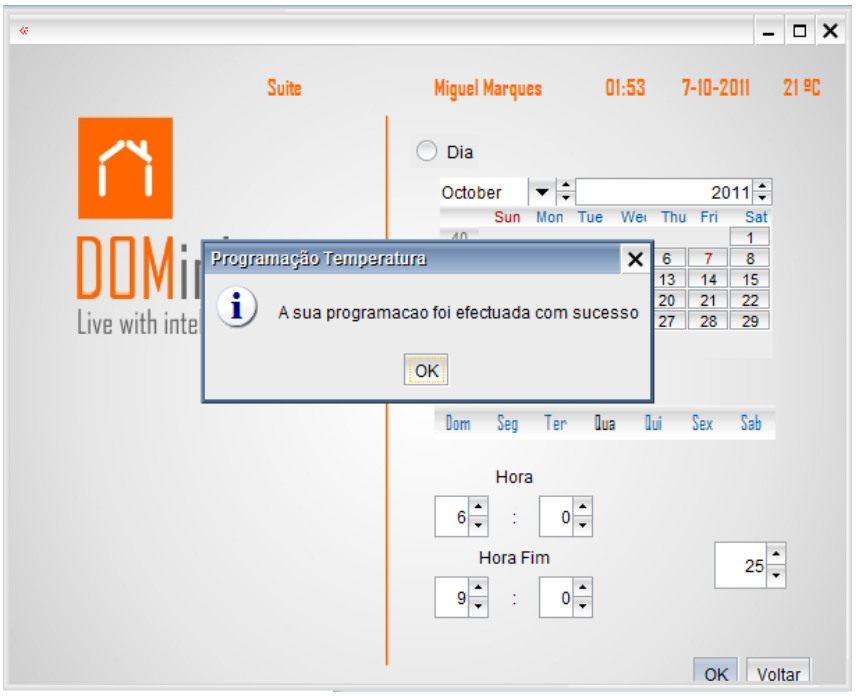


Figura 4.26: Interface regulação temperatura.

```
Numero de entradas na tabela Contexto Temporal = 1

Indice = 1
Data e hora de inicio = 2011-10-07T6:0:0
Data e hora de Fim = 2111-10-07T9:0:0
Segunda-Feira = 1
Terça-Feira = 1
Quarta-Feira = 2
Quinta-Feira = 1
Sexta-Feira = 1
Sabado = 1
Domindo = 1
status = 1
```

Figura 4.27: Contexto temporal.

Através desta figura é possível verificar a data de início e data de fim estipulada pelo utilizador, assim como, a definição que esta programação deve acontecer todas as quartas-feiras. Após o preenchimento desta tabela, a interface define no campo valor da actividade pretendida o valor 25:1. Este valor indica ao software de inferências que foi seleccionada a temperatura de 25º para o contexto temporal com o identificador 2.

4.4.8 Alteração de divisão

Nesta secção, é apresentado o resultado da simulação do deslocamento do utilizador do seu quarto para a sala de estar. Pode verificar-se que a interface actualiza as suas coordenadas geográficas na tabela de interfaces 4.28, gerando uma notificação para o software de inferências. Esta notificação origina a geração de um novo menu de actividades apresentado na figura 4.29.

```
Numero de entradas na tabela Interfaces = 1

Indice = 1
Nome Interface = Interface DOMinho
OIDLocalizacao = 1.3.6.1.3.78.2.2.1.3.5:192.168.2.5:1661
Vc = 6
Vl = 3
Va = 3
IP e Porta de notificações = 127.0.0.1:162
```

Figura 4.28: Actualização de coordenadas na tabela interface.

Constata-se que a localização da interface e a sua temperatura foram alteradas, assim como a disposição das actividades apresentadas. Isto deve-se aos novos valores recebidos pela interface, apresentados em 4.30 e 4.31.

4.4.9 Acesso à actividade ver televisão

Neste divisão, o utilizador acede à actividade 1 (ver televisão). O software inteligente infere a existência de canais e programas favoritos para o utilizador. É assim preenchida a tabela de meta-informação 4.32.

Figura 4.29: Interface sala de estar.

Nesta tabela é possível verificar que ambas as entradas foram geradas para o menu 1 e actividade 1. Na primeira entrada é verificado a existência de um favorito do tipo 1 (canal) que contem os valores "RTP1" e "TVI". Da mesma forma foi inferido que o utilizador não pode visualizar o canal "sportTV". Já a entrada 2 tem associado o favorito do tipo 3 (programa favorito), referente ao programa "Ídolos".

Além da tabela de meta-informação são geradas as actividades descendentes de ver televisão. Desta forma são apresentadas as entradas correspondentes à tabela de actividadesAApresentar. Neste caso, são inferidas as actividades 2 "mudar canal", 3 "desligar TV" e 4 "mudar volume".

Na figura 4.33 pode ser constatado que as actividades são apresentadas por ordem preferencial do utilizador e que as informações recebidas da tabela da meta-informação são também apresentadas ao utilizador.

4.4.10 Estatísticas realizadas

Após todas as interacções efectuadas pelo utilizador são apresentadas as estatísticas geradas pelo módulo de utilizadores. Estas estatísticas têm um valor de confiança atribuído a cada actividade

```
Numero de entradas na tabela Actividades a Apresentar ao utilizador = 5

Indice = 1
Idx Menu = 1
Idx actividade = 1
OID Equipamento = 1.3.6.1.3.78.3.2.1.2.2:192.168.2.5:1661
OID Valor = null
Valor da preferencia para a actividade = 80
OID Localizacao = 1.3.6.1.3.78.2.2.1.3.5:192.168.2.5:1661
Valor Actual da actividade = On
Status = 1

Indice = 2
Idx Menu = 1
Idx actividade = 8
OID Equipamento = 0
OID Valor = null
Valor da preferencia para a actividade = 70
OID Localizacao = 1.3.6.1.3.78.2.2.1.3.5:192.168.2.5:1661
Valor Actual da actividade = null
Status = 1

Indice = 3
Idx Menu = 1
Idx actividade = 5
OID Equipamento = 1.3.6.1.3.78.3.2.1.2.4:192.168.2.5:1661
OID Valor = null
Valor da preferencia para a actividade = 60
OID Localizacao = 1.3.6.1.3.78.2.2.1.3.5:192.168.2.5:1661
Valor Actual da actividade = null
Status = 1
```

Figura 4.30: Actividades a apresentar para a sala de estar.

e têm associadas o contextos nas quais foram realizadas. Ver as tabelas 4.34, 4.35, 4.36, 4.37 e 4.38.

Pode ser verificado que, para a actividade 13 (outras actividades), foi registado um nível de confiança de 20%. Este valor deve-se ao utilizador ter realizado três actividades no período de tempo definido para as estatísticas. Na tabela 4.37, é possível verificar quais os contextos em que o utilizador interagiu com o sistema.

Para a primeira estatística calculada, o contexto de utilizador possui o valor "0" devido à inexistência de qualquer tipo de sistema de localização para utilizadores; o contexto temporal da entrada gerada corresponde ao índice 1 da tabela apresentada em 4.38; a interface possui o identificador 1 e não existe qualquer tipo de equipamento associado a esta entrada.

```
Indice = 4
Idx Menu = 1
Idx actividade = 14
OID Equipamento = 1.3.6.1.3.78.3.2.1.2.4:192.168.2.5:1661
OID Valor = null
Valor da preferencia para a actividade = 50
OID Localizacao = 1.3.6.1.3.78.2.2.1.3.5:192.168.2.5:1661
Valor Actual da actividade = 23
Status = 0

Indice = 5
Idx Menu = 1
Idx actividade = 13
OID Equipamento = 0
OID Valor = null
Valor da preferencia para a actividade = 30
OID Localizacao = 1.3.6.1.3.78.2.2.1.3.5:192.168.2.5:1661
Valor Actual da actividade = null
Status = 1
```

Figura 4.31: Actividades a apresentar para a sala de estar 2.

```
Numero de entradas na tabela Meta-Informação = 2

Indice = 1
idx Menu = 1
idx Actividade = 1
idx Tipo Favorito = 1
Valores = TVI:RTP1
Restrições = SportTV
Status = 1

Indice = 2
idx Menu = 1
idx Actividade = 1
idx Tipo Favorito = 3
Valores = Idolos
Restrições = null
Status = 1
```

Figura 4.32: Meta-informação.

Figura 4.33: Interface ver televisão.

```
Numero de entradas na tabela estatististicas de utilizador = 9

Indice = 1
Idx actividade = 13
OID Valor = 1.3.6.1.3.79.11.2.1.2.10
Valor = 1.3.6.1.3.78.2.2.1.3.4:192.168.2.4:1661
Idx Utilizador = 1
Idx Contexto = 1
nivel de confiança = 20
Status = 1

Indice = 2
Idx actividade = 12
OID Valor = 1.3.6.1.3.79.11.2.1.2.11
Valor = 25:1
Idx Utilizador = 1
Idx Contexto = 2
nivel de confiança = 20
Status = 1

Indice = 3
Idx actividade = 1
OID Valor = 1.3.6.1.3.79.11.2.1.2.12
Valor = c
Idx Utilizador = 1
Idx Contexto = 3
nivel de confiança = 20
Status = 1
```

Figura 4.34: Estatísticas.

```
Indice = 4
Idx actividade = 2
OID Valor = 1.3.6.1.3.79.11.2.1.2.13
Valor = Idolos
Idx Utilizador = 1
Idx Contexto = 3
nivel de confiança = 40
Status = 1

Indice = 5
Idx actividade = 2
OID Valor = 1.3.6.1.3.79.11.2.1.2.14
Valor = Idolos
Idx Utilizador = 1
Idx Contexto = 4
nivel de confiança = 36
Status = 1

Indice = 6
Idx actividade = 2
OID Valor = 1.3.6.1.3.79.11.2.1.2.15
Valor = TVI
Idx Utilizador = 1
Idx Contexto = 4
nivel de confiança = 18
Status = 1
```

Figura 4.35: Estatísticas 2.

Como podemos verificar nesta tabela, o contexto temporal 1 corresponde à data de 7/10/2011 com início às 1:48:19 e teve uma duração de 30 minutos, o que corresponde ao período de tempo programado para as estatísticas.

```
Indice = 7
Idx actividade = 2
OID Valor = 1.3.6.1.3.79.11.2.1.2.16
Valor = RTP1
Idx Utilizador = 1
Idx Contexto = 27
nivel de confiança = 5
Status = 1

Indice = 8
Idx actividade = 3
OID Valor = 1.3.6.1.3.79.9.2.1.2.176
Valor = 51
Idx Utilizador = 1
Idx Contexto = 9
nivel de confiança = 9
Status = 1

Indice = 9
Idx actividade = 4
OID Valor = 1.3.6.1.3.79.11.2.1.2.17
Valor = c
Idx Utilizador = 1
Idx Contexto = 4
nivel de confiança = 9
Status = 1
```

Figura 4.36: Estatísticas 3.

```
Numero de entradas na tabela contexto = 4

Indice = 1
Idx contexto temporal = 1
Contexto Utilizador = 0
OID Localizacao = 1.3.6.1.3.78.2.2.1.3.2:192.168.2.5:1661
OID Equipamento = 0
Idx Interface = 1
status = 1

Indice = 2
Idx contexto temporal = 1
Contexto Utilizador = 0
OID Localizacao = 1.3.6.1.3.78.2.2.1.3.2:192.168.2.5:1661
OID Equipamento = 1.3.6.1.3.78.3.2.1.2.6:192.168.2.5:1661
Idx Interface = 1
status = 1

Indice = 3
Idx contexto temporal = 1
Contexto Utilizador = 0
OID Localizacao = 1.3.6.1.3.78.2.2.1.3.5:192.168.2.5:1661
OID Equipamento = 1.3.6.1.3.78.3.2.1.2.2:192.168.2.5:1661
Idx Interface = 1
status = 1

Indice = 4
Idx contexto temporal = 2
Contexto Utilizador = 0
OID Localizacao = 1.3.6.1.3.78.2.2.1.3.5:192.168.2.5:1661
OID Equipamento = 1.3.6.1.3.78.3.2.1.2.2:192.168.2.5:1661
Idx Interface = 1
status = 1
```

Figura 4.37: Contextos.

```
Numero de entradas na tabela contexto Temporal = 2

Indice = 1
Data/Hora Inicio = 2011-10-7T1:48:19
Data/Hora Fim = 2011-10-7T2:18:19
Segunda-Feira = 1
Terça-Feira = 1
Quarta-Feira = 1
Quarta-Feira = 1
Quinta-Feira = 1
Sexta-Feira = 1
Sabado = 1
Domingo = 1
status = 1

Indice = 2
Data/Hora Inicio = 2011-10-7T2:18:20
Data/Hora Fim = 2011-10-7T2:48:20
Segunda-Feira = 1
Terça-Feira = 1
Quarta-Feira = 1
Quarta-Feira = 1
Quinta-Feira = 1
Sexta-Feira = 1
Sabado = 1
Domingo = 1
status = 1
```

Figura 4.38: Contexto temporal.

Capítulo 5

Conclusão

5.1 Síntese do trabalho e conclusões finais

Nesta dissertação foram apresentados dois módulos desenvolvidos no âmbito do projecto DOMinho, o sistema de utilizadores e o sistema interfaces. Estes módulos visam a concepção de um sistema adaptável automaticamente às preferências, hábitos e comportamentos dos utilizadores.

Este trabalho foi, assim, dividido em duas partes essenciais, uma de investigação e outra de desenvolvimento. Durante a investigação foi notória a falta da sensibilidade das empresas para o fornecimento de interfaces inteligentes, assim como a inexistência de normalização nos protocolos utilizados. Foi no entanto verificado que a sua disponibilização fornece diversas vantagens aos utilizadores, facilitando as suas tarefas diárias e permitindo que pessoas com deficiências possam utilizar mais facilmente este tipo de sistemas.

A utilização de um protocolo normalizado e a modularização deste sistema tem também vantagens, como a simplificada integração de novos serviços, o desenvolvimento de diferentes módulos por entidades diferentes, a fácil integração de módulos num qualquer produto seguindo a mesma arquitectura, entre outros. A definição de uma "comunidade de domótica" neste projecto, faz também, com que as actualizações de novos objectos possam ser facilmente adicionadas aos sistemas sem que seja necessária a sua reestruturação.

O SNMP foi o protocolo comunicacional escolhido para o DOMinho. Tal como constatado em alguns estudos realizados sobre a sua potencialidade neste tipo de sistema, também nos testes

realizados no protótipo desenvolvido foi verificado que a sua integração em domótica tem vantagens e é adequada. A tecnologia sem fios escolhida para transportar este protocolo não pôde ser validada devido à indisponibilidade deste tipo de módulos. No entanto, baseado nos estudos efectuados, espera-se que a tecnologia UWB se adeque ao sistema desenvolvido, pois possibilita mecanismos de localização com elevada acuidade e tempos de resposta muito rápidos e para além disso, possui uma largura de banda elevada, o que permite que serviços com necessidade de maior largura de banda possam ser disponibilizados.

Durante os testes confirmaram-se as esperadas vantagens da utilização de um protocolo e sistema normalizado na domótica, pois permitiu uma rápida e fácil integração de todos os módulos da arquitectura DOMinho. Esta interoperabilidade possibilita que os fabricantes possam especializar-se em diferentes módulos/serviços/equipamentos.

Através da disponibilização dinâmica das informações de preferências, hábitos e comportamentos fornecidas, pelo módulo de utilizadores, foi verificado que é possível a inferência de serviços e interfaces com maior exactidão. Com a atribuição dos contextos temporal, espacial, de utilizador, de equipamento e de interface, foi ainda verificado que estas informações suportam dados suficientemente precisos e flexíveis, sendo fornecido à camada de inferências informações concretas sobre os utilizadores permitindo que serviços e interfaces possam ser disponibilizados com maior nível de eficácia. O módulo de interfaces, por outro lado, permite que novos serviços, características, tipos de interacção e tipos de menus possam ser adicionados ao sistema sem a necessidade de o reestruturar. Permite ainda que interfaces de diferentes níveis de complexibilidade possam interagir com o sistema, possibilitando por exemplo, que o software de inferências possa referir à interface que determinado menu deve ser apresentado com maior tamanho do que outros caso sejam considerados problemas de visão no utilizador. Por outro lado, permite, não só, identificar a forma como o menu deve ser apresentado, mas pode ir a um nível mais baixo, permitindo fornecer informações de apresentação específica para uma determinada actividade presente num menu.

Através dos testes realizados ao protótipo desenvolvido, foi concluído que o modelo possui um bom desempenho, sendo todos as interfaces apresentadas em real-time e todas as adaptações realizadas com sucesso. A utilização de notificações SNMP retira uma parte significativa do pesado processamento nas interfaces. O módulo de utilizadores, por sua vez, realizou correctamente o mapeamento dos estereótipos referentes ao utilizador registado no sistema e efectuou correctamente o cálculo das estatísticas referentes às interacções desse utilizador. Após algum

tempo de testes foi verificado que o sistema fica um pouco mais lento devido à grande quantidade de informação introduzida nas tabelas. No entanto, e caso o número de utilizadores seja relativamente elevado, é possível ultrapassar este problema através da distribuição das informações em mais do que uma tabela do mesmo tipo ou através da utilização de mais do que um sistema de utilizadores e interfaces nos sistema domóticos, uma vez que a arquitectura é completamente modular. Foi também verificado que a primeira interacção realizada, entre o utilizador e o sistema, é relativamente morosa. No entanto, após uma análise mais profunda ao protótipo, foi detectado que este se deve ao modulo de inferências e à forma não optimizada como foi implementado. Foi também verificado que as tabelas criadas possuem uma grande relevância no sistema e que os seus atributos permitem um bom funcionamento do mesmo.

Em relação ao protótipo criado, foi concluído que a API Swing do java, utilizada para o desenvolvimento da parte gráfica, é pouco flexível, tornando difícil o desenvolvimento de interfaces muito complexas.

Os trabalhos relacionados, apresentados no capitulo 2.2 mostraram possuir uma grande relevância e trazer uma grande inovação nos serviços de domótica. No entanto, devido à grande generalidade do sistema DOMinho este pode trazer mais-valias importantes no presente e no futuro, suportando novos serviços e funcionalidades. A sua estrutura modular atribui também uma grande flexibilidade ao sistema, permitindo que cada módulo possa ser desenvolvido por entidades especializadas e distintas. Além disso, a utilização de um protocolo normalizado como o SNMP torna mais simples a interoperabilidade entre sistemas, e permite que o sistema possa estar dividido em diferentes máquinas, reduzindo o peso do processamento.

Para concluir, todos os objectivos inicialmente propostos foram cumpridos, contudo, não foi possível validar e testar o uso do protocolo UWB para a interface, pois não foi possível ter acesso a nenhum módulo UWB.

5.2 Trabalho futuro

Os sistemas desenvolvidos encontram-se ainda numa fase de protótipo, sendo necessária a sua validação num sistema real e com um maior número de serviços. Apesar do sucesso dos testes efectuados, existem ainda melhorias a fazer nos módulos do sistema desenvolvido, tais como:

- efectuar testes em sistemas reais, de forma a detectar eventuais erros e funcionalidades não implementadas;
- evoluir a versão do protocolo SNMP para a versão 3, fornecendo uma maior segurança;
- aprimorar o cálculo do nível de confiança associada às interacções dos utilizadores;
- inserir a noção de contextos na tabela estereótipos;
- alterar a API utilizada para a geração da interface gráfica;
- adicionar técnicas de autenticação baseadas em sensores biométricos;
- adicionar formas de interacção inteligente à interface.

Referências

[1] R. Krish. (2008) Adaptive user interfaces for health care applications. [Online]. Available: http://download.boulder.ibm.com/ibmdl/pub/software/dw/web/wa-uihealth/wa-uihealth-pdf.pdf

[2] DOMINHO. (2011). [Online]. Available: http://www.facebook.com/home.php?#!/pages/DOMinho/143308922378254

[3] F. J. G. J. M. Antonio E. Martinez, Ruben Cabello, "Interact-ddm, a solution for the integration of domestic devices on network management platforms," *IFIP/IEEE International Symposium on Integrated Network Management*, 2003.

[4] Vivimat. (2010). [Online]. Available: http://www.vivimat.com/cas/site/index.php

[5] Control4. (2010). [Online]. Available: http://www.control4.com/

[6] Cardio. (2010). [Online]. Available: www.cardio.pt/

[7] empresa casa do Futuro. (2010). [Online]. Available: http://Casadofuturo.com

[8] casa do Futuro. (2010). [Online]. Available: http://casadofuturo.com/solucao2.asp

[9] Amigo. (2010). [Online]. Available: http://www.hitech-projects.com/euprojects/amigo/

[10] A. Garate, N. Herrasti, and A. Lopez, "Genio: an ambient intelligence application in home automation and entertainment environment," in *Proceedings of the 2005 joint conference on Smart objects and ambient intelligence: innovative context-aware services: usages and technologies*, ser. sOc-EUSAI '05. New York, NY, USA: ACM, 2005, pp. 241–245. [Online]. Available: http://doi.acm.org/10.1145/1107548.1107609

[11] J. R. Velasco, I. M. Maestre, A. Navarro, M. A. Lopez, A. J. Vicente, E. D. L. Hoz, A. Paricio, and M. Machuca, "Location-aware services and interfaces in smart homes using multiagent systems," in *Proc. 2005 Int. Conferecence on Pervasive Systems and Computing. PSC'05. (Las Vegas*, 2005, pp. 1–932 415.

[12] P. Kortum, *The Essential Guide to User Interface Design - An Introduction to GUI Design Principles and Techniques, Second Edition*. John Wiley & Sons, Inc, 2002.

[13] C. Stephanidis, "User interfaces for all: New perspectives into human-computer interaction," *User Interfaces for AllConcepts Methods and Tools*, vol. 1, pp. 3–17, 2001. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.98.4790&rep=rep1&type=pdf

[14] B. Shneiderman, "Direct manipulation for comprehensible, predictable and controllable user interfaces," in *Proceedings of IUI97, 1997 International conference on Intelligent User Interfaces*. ACM Press, 1997, pp. 33–39.

[15] J. A. R. U. B. E. Z. Victor Alvarez-Cortes, Victor H. Zarate, *Advances in Human-Computer Interaction, Current Challenges and Applications for Adaptive User Interfaces*, 1st ed., S. Pinder, Ed. Intellect, 2008.

[16] P. Langley, "User modeling in adaptive interfaces," in *Proceedings of the seventh international conference on User modeling*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1999, pp. 357–370. [Online]. Available: http://dl.acm.org/citation.cfm?id=317328.317406

[17] G. S. Yong G. Ji, "A framework for improving organizational learning through a user-adaptive intranet portal organizational memory information system," *The International Journal of Aviation Psychology*, vol. Volume 11, Issue 2, pp. 123–148, 2001.

[18] A. Granic, *Advances in Human-Computer Interaction Edited, Intelligent Interfaces for Technology-Enhanced Learning*, 1st ed. Intellect, 2008.

[19] A. F. Norcio and J. Stanley, "Adaptive human-computer interfaces: a literature survey and perspective," *Ieee Transactions On Systems Man And Cybernetics*, vol. 19, no. 2, pp. 399–408, 1989. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=31042

[20] L. J. H. Michael W. Haas, "Current research in adaptive interfaces," *The International Journal of Aviation Psychology*, vol. Volume 11, Issue 2, pp. 399–408, 2001.

[21] A. Kobsa, "Supporting user interfaces for all through user modeling," *Advances in Human FactorsErgonomics*, vol. 20, pp. 155–157, 1995. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0921264706800274

[22] V. Damjanovic and M. Kravcik, *Using Emotional Intelligence in Personalized Adaptation*. IGI Publishing, 2008, pp. 1716–1742. [Online]. Available: http://hdl.handle.net/1820/1121

[23] H. Dieterich, U. Malinowski, T. Kuhme, and M. Schneider-Hufschmidt, "State of the art in adaptive user interfaces," in *Adaptive User Interfaces: Principles and Practice*, M. Schneider-Hufschmidt, T. Kuhme, and U. Malinowski, Eds. Amsterdam: North-Holland, 1993, pp. 13–48.

[24] J. A. R. U. Victor Alvarez-Cortes, Victor H. Zarate and B. E. Zayas, *Principles of interactive multimedia*, 1st ed., M.-H. E. Europe, Ed. McGraw-Hill, 2000.

[25] Z. L. C. M. G. S. Nikos Tsianos, Panagiotis Germanakos, *Intelligent User Interfaces: Adaptation and Personalization Systems and Technologies, An Assessment of Human Factors in Adaptive Hypermedia Environments*, 1st ed. Morgan Kaufmann, 2009.

[26] D. Benyon and D. Murray, "Applying user modeling to human-computer interaction design," *Artificial Intelligence Review*, vol. 7, no. 3-4, pp. 199–225, 1993. [Online]. Available: http://www.springerlink.com/index/10.1007/BF00849555

[27] J. Pan, B. Zhang, and G. Wu, "A sam-based evolution model of ontological user model," in *Proceedings of the 2009 Eigth IEEE/ACIS International Conference on Computer and Information Science*. Washington, DC, USA: IEEE Computer Society, 2009, pp. 1139–1143. [Online]. Available: http://dl.acm.org/citation.cfm?id=1605395.1605858

[28] L. Nguyen and P. Do, "Learner model in adaptive learning," *World Academy of Science Engineering and Technology*, vol. 45, pp. 395–400, 2008. [Online]. Available: http://www.waset.org/journals/waset/v45/v45-70.pdf

[29] ACM. (2010). [Online]. Available: http://www.acm.org/crossroads/xrds3-3/haptic.html

[30] G. C. Burdea, "Haptic feedback for virtual reality," *Virtual Reality and Prototyping Workshop*, vol. 2, no. June, pp. 17–29, 1999. [Online]. Available: http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Haptic+Feedback+for+Virtual+Reality#0

[31] A. G. Marcia K. O'Malley, *HCI beyond the GUI: Design for Haptic, Speech, Olfactory, and Other Non Traditional Interfaces, Haptic Interfaces*, 1st ed. Morgan Kaufmann, 2008.

[32] A. Cockburn and S. Brewster, "Multimodal feedback for the acquisition of small targets," 2005.

[33] Samsung. (2010). [Online]. Available: http://www.samsung.com/hk_en/news/newsRead.do?news_group=productnews&news_type=consumerproduct&news_ctgry=mobilephone&news_seq=9140

[34] Veja.com. (2010). [Online]. Available: http://veja.abril.com.br/160408/p_100.shtml

[35] iRoboticist. (2010) iroboticist. [Online]. Available: http://iroboticist.com/2010/03/26/tgv/

[36] I. Poupyrev, S. Maruyama, and J. Rekimoto, "Ambient touch: designing tactile interfaces for handheld devices," in *Proceedings of the 15th annual ACM symposium on User interface software and technology*, ser. UIST '02. New York, NY, USA: ACM, 2002, pp. 51–60. [Online]. Available: http://doi.acm.org/10.1145/571985.571993

[37] K. Yatani and K. N. Truong, "Semfeel: a user interface with semantic tactile feedback for mobile touch-screen devices," in *Proceedings of the 22nd annual ACM symposium on User interface software and technology*, ser. UIST '09. New York, NY, USA: ACM, 2009, pp. 111–120. [Online]. Available: http://doi.acm.org/10.1145/1622176.1622198

[38] M. Hafez, "Tactile interfaces: technologies, applications and challenges," vol. 23, no. 4. Secaucus, NJ, USA: Springer-Verlag New York, Inc., March 2007, pp. 267–272. [Online]. Available: http://www.springerlink.com/index/10.1007/s00371-007-0102-2

[39] E. Hoggan, S. A. Brewster, and J. Johnston, "Investigating the effectiveness of tactile feedback for mobile touchscreens," in *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, ser. CHI '08. New York, NY, USA: ACM, 2008, pp. 1573–1582. [Online]. Available: http://doi.acm.org/10.1145/1357054.1357300

[40] T. Koskela and K. Vaananen-Vainio-Mattila, "Evolution towards smart home environments: empirical evaluation of three user interfaces," *Personal Ubiquitous Comput.*, vol. 8, pp. 234–240, July 2004. [Online]. Available: http://dx.doi.org/10.1007/s00779-004-0283-x

[41] A. Ibrahim, J. Lundberg, and J. Johansson, "Speech enhanced remote control for media terminal," in *In: Proceedings of Eurospeech '01. Volume 4*, 2001, pp. 2685–2688.

[42] B. Brumitt, J. Cadiz, and J. Cadiz, ""let there be light!- comparing interfaces for homes of the future," 2000.

[43] M. Gandy, T. Starner, J. Auxier, and D. Ashbrook, "The gesture pendant: A self-illuminating, wearable, infrared computer vision system for home automation control and medical monitoring," in *Proceedings of the 4th IEEE International Symposium on Wearable Computers*, ser. ISWC '00. Washington, DC, USA: IEEE Computer Society, 2000, pp. 87–94. [Online]. Available: http://dl.acm.org/citation.cfm?id=851037.856538

[44] I. Potamitis, K. Georgila, N. Fakotakis, and G. Kokkinakis, "An integrated system for smart-home control of appliances based on remote speech interaction," 2003.

[45] H. Habib and M. Mufti, "Real time mono vision gesture based virtual keyboard system," *IEEE Transactions on Consumer Electronics*, vol. 52, no. 4, pp. 1261–1266, 2006. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4050054

[46] M. Karam, J. C. Lee, T. Rose, F. Quek, and S. McCrickard, "Comparing gesture and touch for notification system interactions," in *Proceedings of the 2009 Second International Conferences on Advances in Computer-Human Interactions*, ser. ACHI '09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 7–12. [Online]. Available: http://dx.doi.org/10.1109/ACHI.2009.65

[47] D. M. Krum, O. Omoteso, W. Ribarsky, T. Starner, and L. F. Hodges, "Speech and gesture multimodal control of a whole earth 3d visualization environment," in *Proceedings of the symposium on Data Visualisation 2002*, ser. VISSYM '02. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2002, pp. 195–200. [Online]. Available: http://dl.acm.org/citation.cfm?id=509740.509772

[48] L.-P. Morency and T. Darrell, "Head gesture recognition in intelligent interfaces: the role of context in improving recognition," in *Proceedings of the 11th international conference on Intelligent user interfaces*, ser. IUI '06. New York, NY, USA: ACM, 2006, pp. 32–38. [Online]. Available: http://doi.acm.org/10.1145/1111449.1111464

[49] S. O. Paulo Barthelmess, *HCI beyond the GUI: Design for Haptic, Speech, Olfactory, and Other Non Traditional Interfaces, Multimodal Interfaces: Combining Interfaces to Accomplish a Single Task*, 1st ed. Morgan Kaufmann, 2008.

[50] Z. Callejas and R. Lopez-Cozar, "Designing smart home interfaces for the elderly," *SIGACCESS Access. Comput.*, pp. 10–16, September 2009. [Online]. Available: http://doi.acm.org/10.1145/1651259.1651261

[51] E. van den Hoven, J. Frens, D. Aliakseyeu, J.-B. Martens, K. Overbeeke, and P. Peters, "Design research & tangible interaction," in *Proceedings of the 1st international conference on Tangible and embedded interaction*, ser. TEI '07. New York, NY, USA: ACM, 2007, pp. 109–115. [Online]. Available: http://doi.acm.org/10.1145/1226969.1226993

[52] J. W. Frens, *Designing for Rich Interaction: Integrating Form, Interaction, and Function*, 1st ed. Johannes Willem Frens, 2006.

[53] F. Block, A. Schmidt, N. Villar, and H.-W. Gellersen, "Towards a playful user interface for home entertainment systems." in *EUSAI*, ser. Lecture Notes in Computer Science, P. Markopoulos, B. Eggen, E. H. L. Aarts, and J. L. Crowley, Eds., vol. 3295. Springer, 2004, pp. 207–217. [Online]. Available: http://dblp.uni-trier.de/db/conf/eusai/eusai2004.html#BlockSVG04

[54] C. Forlines, D. Wigdor, C. Shen, and R. Balakrishnan, "Direct-touch vs. mouse input for tabletop displays," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, ser. CHI '07. New York, NY, USA: ACM, 2007, pp. 647–656. [Online]. Available: http://doi.acm.org/10.1145/1240624.1240726

[55] M. Hall, E. Hoggan, and S. Brewster, "T-bars: towards tactile user interfaces for mobile touchscreens," in *Proceedings of the 10th international conference on Human computer interaction with mobile devices and services*, ser. MobileHCI

'08. New York, NY, USA: ACM, 2008, pp. 411–414. [Online]. Available: http://doi.acm.org/10.1145/1409240.1409301

[56] J. W. C. Kwang Yeol Lee, "Remote-controlled home automation system via bluetooth home network," 2003.

[57] BluetoothSIG. (2010) Bluetooth. [Online]. Available: http://www.bluetooth.com

[58] J.-S. Lee, Y.-W. Su, and C.-C. Shen, "A comparative study of wireless protocols: Bluetooth, uwb, zigbee, and wi-fi," *IECON 2007 33rd Annual Conference of the IEEE Industrial Electronics Society*, pp. 46–51, 2007. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4460126

[59] S. S. D. L. Jeff Foerster, Evan Green, "Ultra-wideband technology for short-or medium-range wireless communications," *Intel Technology Journal*, 2001.

[60] Z. Alliance. (2010) Zigbee. [Online]. Available: http://www.zigbee.org/

[61] X. Liang and I. Balasingham, "Performance analysis of the ieee 802.15. 4 based ecg monitoring network," *Proc The Seventh IASTED International Conferences on Wireless and Optical Communications WOC'07*, pp. 99–104, 2007. [Online]. Available: http://www.actapress.com/abstract.aspx?paperid=30457

[62] A. C. Bhavneet Sidhu, Hardeep Singh, "Emerging wireless standards - wifi, zigbee and wimax," 2007.

[63] F. S. A. L. K. Shuaib, M. Boulmalf, "Co-existence of zigbee and wlan, a performance study," 2006.

[64] M. Petrova, J. Riihijarvi, P. Mahonen, and S. Labella, "Performance study of ieee 802.15.4 using measurements and simulations," *IEEE Wireless Communications and Networking Conference, 2006. WCNC 2006*, vol. 1, pp. 487–492, 2006. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1683512

[65] C. F. Chiasserini and R. R. Rao, "Performance of ieee 802.11 wlans in a bluetooth environment," in *IEEE Wireless Communications and Networking Conference 2000 (WCNC 2000*, 2000, pp. 1–94.

[66] H. Lin, W. Chu, T. Gong, Y. Ti, Y. Sun, J. H. Nielsen, and A. Naseem, "Integrating blip into location-aware system: A service-oriented method," *Convergence Information Technology, International Conference on*, vol. 0, pp. 144–148, 2009.

[67] H. Liu, H. Darabi, P. Banerjee, and J. Liu, "Survey of wireless indoor positioning techniques and systems." *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, pp. 1067–1080, 2007.

[68] J. H. Krzysztof W. Kolodziej, *Local Positioning Systems LBS Applications and Services: Existing Indoor Location Systems: How They Work*, 1st ed. CRC/Taylor & Francis, 2006.

[69] E. M. David T. Perkins, *Understanding SNMP MIBs*, 1993.

[70] G. S. Nikolay Kakanakov, Elena Kostadinova, "Using snmp for remote measurement and automation," *Electronics*, 2007.

[71] K. J. Hongseok Jang and D. Choi, "Delivery and storage architecture for sensing information using snmp," *IJCSNS International Journal of Computer Science and Network Security, VOL.6 No.4, April 2006*, vol. 6, no. 4, 2006.

[72] S. I. Demeter Robert, "Snmp protocol based home automation system," *Roedunet International Conference (RoEduNet), 2011 10th*, 2011.

[73] SNMP4j. (2010). [Online]. Available: http://www.snmp4j.org/

[74] AgenPro3. (2010). [Online]. Available: http://www.agentpp.com/agen/agen.html

MIB utilizadores

```
DomoticaSNMPUsers-MIB DEFINITIONS ::= BEGIN

IMPORTS

MODULE-IDENTITY, OBJECT-TYPE, NOTIFICATION-TYPE, Counter64,
Unsigned32, Integer32, private, Opaque, TimeTicks, snmpModules
FROM SNMPv2-SMI
DisplayString, DateAndTime, RowStatus, StorageType, VariablePointer
FROM SNMPv2-TC
SnmpAdminString
FROM SNMP-FRAMEWORK-MIB
MODULE-COMPLIANCE, OBJECT-GROUP, NOTIFICATION-GROUP
FROM SNMPv2-CONF;


--domoticExp    OBCECT IDENTIFIER ::= { private 1 }

domoticaSNMPMibUsers MODULE-IDENTITY
LAST-UPDATED            "201006290000Z"
ORGANIZATION            "DOMinho"
CONTACT-INFO
"Marino Fernandes
email: marinofernandes@hotmail.com"
DESCRIPTION
"MIB para Utilizador"
REVISION                        "201006290000Z"
DESCRIPTION                     "MIB Utilizadores"
::= {experimental 79}


--groups in DOMOTICEXP

system
OBJECT IDENTIFIER ::= { domoticaSNMPMibUsers 1 }
user
OBJECT IDENTIFIER ::= { domoticaSNMPMibUsers 2 }
userProfileValues
OBJECT IDENTIFIER ::= { domoticaSNMPMibUsers 3 }
```

```
favoritesInferences
OBJECT IDENTIFIER ::= { domoticaSNMPMibUsers 4 }
inferencesUserActivities
OBJECT IDENTIFIER ::= { domoticaSNMPMibUsers 5 }
statisticsUserActivities
OBJECT IDENTIFIER ::= { domoticaSNMPMibUsers 6 }
context
OBJECT IDENTIFIER ::= { domoticaSNMPMibUsers 7 }
temporalContext
OBJECT IDENTIFIER ::= { domoticaSNMPMibUsers 8 }
intValues
OBJECT IDENTIFIER ::= { domoticaSNMPMibUsers 9 }
realValue
OBJECT IDENTIFIER ::= { domoticaSNMPMibUsers 10 }
stringValue
OBJECT IDENTIFIER ::= { domoticaSNMPMibUsers 11 }
stereotypeProfile
OBJECT IDENTIFIER ::= { domoticaSNMPMibUsers 12 }
authentication
OBJECT IDENTIFIER ::= { domoticaSNMPMibUsers 13 }
complianceGroup
OBJECT IDENTIFIER ::= { domoticaSNMPMibUsers 14 }
notificationInterface
OBJECT IDENTIFIER ::= { domoticaSNMPMibUsers 15 }

-- SYSTEM (1)*****************************************

sysDescr OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"Descricao textual da entidade. Inclui informacoes
como nome, versao do software utilizado..."
::= { system 1 }

sysContact OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"Descricao textual do contacto da pessoa que
desenvolveu o sistema."
::= { system 2 }

sysName OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-only
STATUS current
```

```
DESCRIPTION
"Descricao textual com o nome do sistema."
::= { system 3 }

sysLocation OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"Localizacao fisica do sistema. Se a localizacao for
desconhecida, a string fica vazia."
::= { system 4 }

--User (2)**************************************

userNumber OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"Numero de utilizadores existentes no sistema."
::= { user 1 }

userTable OBJECT-TYPE
SYNTAX SEQUENCE OF UserEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Uma lista com o conjunto de utilizadores que fazem
parte do sistema, correspondendo uma entrada a cada
utilizador."
::= { user 2 }

userEntry OBJECT-TYPE
SYNTAX UserEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Uma entrada que contem informacoes de cada utilizador"
INDEX { userIndex }
::= { userTable 1 }

UserEntry ::= SEQUENCE {
userIndex                    Unsigned32,
userDesignation              DisplayString,
typeUser                     Unsigned32,
oidLocationUser              DisplayString,
vc                           DisplayString,
vl                           DisplayString,
```

```
va                              DisplayString,
statusUser                      Unsigned32
}

userIndex OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"Um unico valor maior do que zero para cada novo
utilizador. Os valores atribuidos variam entre 1
e N utilizadores."
::= { userEntry 1 }

userDesignation OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"Nome do utilizador."
::= { userEntry 2 }

typeUser OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"tipo de utilizador."
::= { userEntry 3 }

oidLocationUser OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"Localizacao do utilizador no sistema"
::= { userEntry 4 }

vc OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"comprimento do vector da localizacao do utilizador "
::= { userEntry 5 }

vl OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-write
```

```
STATUS current
DESCRIPTION
"largura do vector da localizacao do utilizador."
::= { userEntry 6 }

va OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"altura do vector da localizacao do utilizador."
::= { userEntry 7 }

statusUser OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"status da entrada. permite por exemplo que a entrada
seja apagada"
::= { userEntry 8 }

-- UserProfileValues (3)*****************************

userProfileValuesNumber OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"Numero de perfis de utilizador existentes neste sistema."
::= { userProfileValues 1 }

userProfileValuesTable OBJECT-TYPE
SYNTAX SEQUENCE OF UserProfileValuesEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Lista com os varios perfis."
::= { userProfileValues 2 }

userProfileValuesEntry OBJECT-TYPE
SYNTAX UserProfileValuesEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Cada entrada contem informacoes de um perfil de
utilizador especifico."
INDEX {userProfileValuesIndex}
::= {userProfileValuesTable 1}
```

```
UserProfileValuesEntry ::= SEQUENCE {
userProfileValuesIndex                  Unsigned32,
idxUser                                 Unsigned32,
idxUserProfileComunity                  Unsigned32,
oidValueUserProfile                     VariablePointer,
confidenceLevelUserProfile              Unsigned32,
idxContextUserProfile                   Unsigned32,
statusUserProfile                                          Unsigned32
}

userProfileValuesIndex OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada novo perfil.
Os valores atribuidos devem variar entre 1 e N."
::= { userProfileValuesEntry 1 }

idxUser OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identificador do utilizador ao qual esta associado
o perfil."
::= { userProfileValuesEntry 2 }

idxUserProfileComunity OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identificador do perfil definido pela comunidade
de domotica."
::= { userProfileValuesEntry 3 }


oidValueUserProfile OBJECT-TYPE
SYNTAX VariablePointer
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"OID da tabela de valores onde pode ser encontrado o
valor do perfil associado ao utilizador."
::= { userProfileValuesEntry 4 }

confidenceLevelUserProfile OBJECT-TYPE
```

```
SYNTAX Unsigned32 (0..100)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"nivel de confianca atribuido a entrada."
::= { userProfileValuesEntry 5 }

idxContextUserProfile OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"contexto atribuido ao perfil."
::= { userProfileValuesEntry 6 }


statusUserProfile OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"status da entrada. permite por exemplo que a entrada
seja apagada."
::= { userProfileValuesEntry 7 }

-- favoritesInferences (4)*************************

favoritesInferencesNumber OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"Numero de inferias de favoritos existentes no sistema."
::= { favoritesInferences 1 }

favoritesInferencesTable OBJECT-TYPE
SYNTAX SEQUENCE OF FavoritesInferencesEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Lista com as varios favoritos existentes."
::= { favoritesInferences 2 }

favoritesInferencesEntry OBJECT-TYPE
SYNTAX FavoritesInferencesEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Cada entrada contem informacoes de favoritos aplicaveis
```

```
inferidos."
INDEX {favoritesInferencesIndex}
::= {favoritesInferencesTable 1}

FavoritesInferencesEntry ::= SEQUENCE {
favoritesInferencesIndex            Unsigned32,
idxTypeFavorites                    Unsigned32,
idxUserFavoritesInferences          Unsigned32,
idxContextFavoritesInferences       Unsigned32,
oidValueFavoritesInferences         VariablePointer,
confidenceLevelFavoritesInferences  Unsigned32,
statusFavorites                                    Unsigned32
}


favoritesInferencesIndex OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada nova inferencia de
favorito. Os valores atribuidos devem variar entre 1 e N."
::= { favoritesInferencesEntry 1 }

idxTypeFavorites OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identificador da Inferencia de favorito definida
pela comunidade."
::= { favoritesInferencesEntry 2 }


idxUserFavoritesInferences OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identificador do utilizador para o qual foi inserido
o favorito."
::= { favoritesInferencesEntry 3 }

idxContextFavoritesInferences OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"contexto para o qual o favorito foi inferido."
```

```
::= { favoritesInferencesEntry 4 }


oidValueFavoritesInferences OBJECT-TYPE
SYNTAX VariablePointer
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"OID da tabela de valores onde pode ser encontrado
o valor do perfil
associado ao utilizador."
::= { favoritesInferencesEntry 5 }

confidenceLevelFavoritesInferences OBJECT-TYPE
SYNTAX Unsigned32 (0..100)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"nivel de confianca atribuida ao favorito."
::= { favoritesInferencesEntry 6 }

statusFavorites OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"status da entrada. permite por exemplo que a entrada
seja apagada."
::= { favoritesInferencesEntry 7 }


-- inferencesUserActivities (5)**********************


inferencesUserActivitiesNumber OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"Numero de estatisticas existentes na tabela."
::= { inferencesUserActivities 1 }

inferencesUserActivitiesTable OBJECT-TYPE
SYNTAX SEQUENCE OF InferencesUserActivitiesEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Lista com as varias estatisticas."
::= { inferencesUserActivities 2 }
```

```
inferencesUserActivitiesEntry OBJECT-TYPE
SYNTAX InferencesUserActivitiesEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Cada entrada contem informacoes de cada inferencia
realizada."
INDEX {inferencesUserActivitiesIndex}
::= {inferencesUserActivitiesTable 1}

InferencesUserActivitiesEntry ::= SEQUENCE {
inferencesUserActivitiesIndex             Unsigned32,
idxActivityInferencesUserActivities       Unsigned32,
oidValueInferencesUserActivities          VariablePointer,
idxUserInferencesUserActivities           Unsigned32,
idxContextInferencesUserActivities        Unsigned32,
confidenceLevelInferencesUserActivities   Unsigned32,
statusUserInferences                                Unsigned32
}

inferencesUserActivitiesIndex OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada nova inferencia.
Os valores atribuidos devem variar entre 1 e N."
::= { inferencesUserActivitiesEntry 1 }

idxActivityInferencesUserActivities OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identifica a actividade correspondente a inferencia."
::= { inferencesUserActivitiesEntry 2 }


oidValueInferencesUserActivities OBJECT-TYPE
SYNTAX VariablePointer
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"OID da tabela de valores onde pode ser encontrado o
valor da inferencia."
::= { inferencesUserActivitiesEntry 3 }
```

```
idxUserInferencesUserActivities OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identifica o utilizador para o qual se refere a
inferencia."
::= { inferencesUserActivitiesEntry 4 }

idxContextInferencesUserActivities OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identifica o contexto para o qual a inferencia foi
calcula."
::= { inferencesUserActivitiesEntry 5 }


confidenceLevelInferencesUserActivities OBJECT-TYPE
SYNTAX Unsigned32 (0..100)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"nivel de confianca atribuido a inferencia."
::= { inferencesUserActivitiesEntry 6 }


statusUserInferences OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"status da entrada. permite por exemplo que a entrada
seja apagada."
::= { inferencesUserActivitiesEntry 7 }


-- statisticsUserActivities (6)****************


statisticsUserActivitiesNumber OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"Numero de estatisticas existentes na tabela."
::= { statisticsUserActivities 1 }
```

```
statisticsUserActivitiesTable OBJECT-TYPE
SYNTAX SEQUENCE OF StatisticsUserActivitiesEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Lista com as varias estatisticas."
::= { statisticsUserActivities 2 }

statisticsUserActivitiesEntry OBJECT-TYPE
SYNTAX StatisticsUserActivitiesEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Cada entrada contem informacoes de cada estatistica
realizada."
INDEX {statisticsUserActivitiesIndex}
::= {statisticsUserActivitiesTable 1}

StatisticsUserActivitiesEntry ::= SEQUENCE {
statisticsUserActivitiesIndex              Unsigned32,
idxActivitystatisticsUserActivities        Unsigned32,
oidValuestatisticsUserActivities           VariablePointer,
idxUserstatisticsUserActivities            Unsigned32,
idxContextstatisticsUserActivities         Unsigned32,
confidenceLevelstatisticsUserActivities    Unsigned32,
statusUserstatistics                                          Unsigned32
}

statisticsUserActivitiesIndex OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada nova estatistica.
Os valores atribuidos devem variar entre 1 e N."
::= { statisticsUserActivitiesEntry 1 }

idxActivitystatisticsUserActivities OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identifica a actividade correspondente a estatistica."
::= { statisticsUserActivitiesEntry 2 }


oidValuestatisticsUserActivities OBJECT-TYPE
SYNTAX VariablePointer
MAX-ACCESS read-create
```

```
STATUS current
DESCRIPTION
"OID da tabela de valores onde pode ser encontrado o
valor da estatistica."
::= { statisticsUserActivitiesEntry 3 }


idxUserstatisticsUserActivities OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identifica o utilizador para o qual se refere a estatistica."
::= { statisticsUserActivitiesEntry 4 }

idxContextstatisticsUserActivities OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identifica o contexto para o qual a estatistica foi calcula."
::= { statisticsUserActivitiesEntry 5 }


confidenceLevelstatisticsUserActivities OBJECT-TYPE
SYNTAX Unsigned32 (0..100)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"nivel de confianca atribuido a estatisticas."
::= { statisticsUserActivitiesEntry 6 }


statusUserstatistics OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"status da entrada. permite por exemplo que a entrada seja
apagada."
::= { statisticsUserActivitiesEntry 7 }

-- Context (7)*****************************************

contextNumber OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-write
STATUS current
DESCRIPTION
```

```
"Numero de contextos existentes."
::= { context 1 }

contextTable OBJECT-TYPE
SYNTAX SEQUENCE OF ContextEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Lista com os varios contextos."
::= { context 2 }

contextEntry OBJECT-TYPE
SYNTAX ContextEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Cada entrada contem informacoes sobre os contextos."
INDEX {contextIndex}
::= {contextTable 1}

ContextEntry ::= SEQUENCE {
contextIndex                    Unsigned32,
idxTemporalContext              Unsigned32,
userContext                     DisplayString,
oidlocation                     DisplayString,
oidEquipmentUsed                DisplayString,
idxInterfaceContext                             Unsigned32,
statusContext                   Unsigned32
}

contextIndex OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada novo contexto.
Os valores atribuidos devem variar entre 1 e N."
::= { contextEntry 1 }

idxTemporalContext OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identifica o contexto temporal."
::= { contextEntry 2 }


userContext OBJECT-TYPE
```

```
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"contexto de utilizador."
::= { contextEntry 3 }

oidlocation OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identifica a localizacao ao qual pertence o contexto."
::= { contextEntry 4 }


oidEquipmentUsed OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identifica o equipamento do contexto."
::= { contextEntry 5 }

idxInterfaceContext OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identifica a interface."
::= { contextEntry 6 }

statusContext OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"status da entrada. permite por exemplo que a entrada
seja apagada."
::= { contextEntry 7 }


-- temporalContext (8)**************************

temporalContextNumber OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-write
STATUS current
DESCRIPTION
```

```
"Numero de contextos temporais existentes na tabela."
::= { temporalContext 1 }

temporalContextTable OBJECT-TYPE
SYNTAX SEQUENCE OF TemporalContextEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Lista com os varios contextos temporais."
::= { temporalContext 2 }

temporalContextEntry OBJECT-TYPE
SYNTAX TemporalContextEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"entradas com o contexto temporal."
INDEX {temporalContextIndex}
::= {temporalContextTable 1}

TemporalContextEntry ::= SEQUENCE {
temporalContextIndex                Unsigned32,
beginDateTime                       DisplayString,
endDateTime                         DisplayString,
monday                                          Unsigned32,
tuesday                                       Unsigned32,
wednesday                           Unsigned32,
thursday                            Unsigned32,
friday                              Unsigned32,
saturday                            Unsigned32,
sunday                              Unsigned32,
statusTemporalContext                           Unsigned32
}

temporalContextIndex OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada novo
contexto temporal. Os valores atribuidos devem
variar entre 1 e N."
::= { temporalContextEntry 1 }

beginDateTime OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
```

```
"data e hora de inicio do contexto."
::= { temporalContextEntry 2 }


endDateTime OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"data e hora de fim do contexto."
::= { temporalContextEntry 3 }


monday OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"todas as segundas-feiras."
::= { temporalContextEntry 4 }

tuesday OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"todas as tercas-feiras."
::= { temporalContextEntry 5 }

wednesday OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"todas as quartas-feiras."
::= { temporalContextEntry 6 }


thursday OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"todas as quintas-feiras."
::= { temporalContextEntry 7 }

friday OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
```

```
STATUS current
DESCRIPTION
"todas as sextas-feiras."
::= { temporalContextEntry 8 }

saturday OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"todas os sabados."
::= { temporalContextEntry 9 }

sunday OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"todas as domingos."
::= { temporalContextEntry 10 }

statusTemporalContext OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"status da entrada. permite por exemplo que a
entrada seja apagada."
::= { temporalContextEntry 11 }


-- intValues (9)*******************************

intValuesNumber OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"Numero de valores inteiros existentes na tabela."
::= {intValues 1 }


intValuesTable OBJECT-TYPE
SYNTAX SEQUENCE OF IntValuesEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Uma lista com o conjunto de valores inteiros."
::= { intValues 2 }
```

```
intValuesEntry OBJECT-TYPE
SYNTAX IntValuesEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Uma entrada que contem informacoes dos inteiros"
INDEX {intValuesIndex}
::= { intValuesTable 1}

IntValuesEntry ::= SEQUENCE {
intValuesIndex          Unsigned32,
valueInt                Integer32,
statusInt                                Unsigned32
}

intValuesIndex OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada novo
valor. Os valores atribuidos devem variam
entre 1 e N."
::= { intValuesEntry 1 }

valueInt OBJECT-TYPE
SYNTAX Integer32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"valor."
::= { intValuesEntry 2 }

statusInt OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"status da entrada. permite por exemplo que a
entrada seja apagada."
::= { intValuesEntry 3 }

-- realValue (10)*****************************

realValueNumber OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-write
STATUS current
```

```
DESCRIPTION
"Numero de valores reais diferentes existentes
neste sistema."
::= {realValue 1 }


realValueTable OBJECT-TYPE
SYNTAX SEQUENCE OF RealValueEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Uma lista com o conjunto de valores reais."
::= { realValue 2 }

realValueEntry OBJECT-TYPE
SYNTAX RealValueEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Uma entrada que contem informacoes de cada entrada"
INDEX {realValueIndex}
::= { realValueTable 1}

RealValueEntry ::= SEQUENCE {
realValueIndex             Unsigned32,
valueReal                  DisplayString,
statusReal                     Unsigned32
}

realValueIndex OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada novo
valor. Os valores
atribuidos devem variam entre 1 e N."
::= { realValueEntry 1 }

valueReal OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"valor."
::= { realValueEntry 2 }

statusReal OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
```

```
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"status da entrada. permite por exemplo que
a entrada seja apagada."
::= { realValueEntry 3 }

-- stringValue (11)************************

stringValueNumber OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada novo
valor. Os valores
atribuidos devem variam entre 1 e N."
::= {stringValue 1 }


stringValueTable OBJECT-TYPE
SYNTAX SEQUENCE OF StringValueEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Uma lista com o conjunto de valores do tipo
string."
::= { stringValue 2 }

stringValueEntry OBJECT-TYPE
SYNTAX StringValueEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Uma entrada que contem informacoes de cada
entrada"
INDEX {stringValueIndex}
::= { stringValueTable 1}

StringValueEntry ::= SEQUENCE {
stringValueIndex            Unsigned32,
valueString                 DisplayString,
statusString                    Unsigned32
}

stringValueIndex OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
```

```
DESCRIPTION
"Um unico valor maior que zero para cada novo
valor. Os valores atribuidos devem variam
entre 1 e N."
::= { stringValueEntry 1 }

valueString OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"valor."
::= { stringValueEntry 2 }

statusString OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"status da entrada. permite por exemplo que a
entrada seja apagada."
::= { stringValueEntry 3 }

-- stereotypeProfile (12)********************


stereotypeProfileNumber OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"Numero de esteriotipos existentes na tabela."
::= { stereotypeProfile 1 }

stereotypeProfileTable OBJECT-TYPE
SYNTAX SEQUENCE OF StereotypeProfileEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Uma lista com varios esteriotipos."
::= { stereotypeProfile 2 }

stereotypeProfileEntry OBJECT-TYPE
SYNTAX StereotypeProfileEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Uma entrada que contem informacoes de cada
esteriotipo."
```

```
INDEX { stereotypeProfileIndex }
::= { stereotypeProfileTable 1 }

StereotypeProfileEntry ::= SEQUENCE {
stereotypeProfileIndex              Unsigned32,
idxStereotype                       Unsigned32,
idxUserProfileStereotype            Unsigned32,
oidValueStereotypeProfile           VariablePointer,
confidenceLevelStereotypeProfile    Unsigned32,
statusStereotypeProfile             Unsigned32
}

stereotypeProfileIndex OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada novo
esteriotipo. Os valores
atribuidos devem variam entre 1 e N."
::= { stereotypeProfileEntry 1 }

idxStereotype OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"identifica o esteriotipo definido pela comunidade."
::= { stereotypeProfileEntry 2 }

idxUserProfileStereotype OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"identifica o perfil de utilizador definido pela
comunidade."
::= { stereotypeProfileEntry 3 }

oidValueStereotypeProfile OBJECT-TYPE
SYNTAX VariablePointer
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"OID da tabela de valores onde pode ser encontrado
o valor da inferencia."
::= { stereotypeProfileEntry 4 }

confidenceLevelStereotypeProfile OBJECT-TYPE
```

```
SYNTAX Unsigned32 (0..100)
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"nivel de confianca atribuido a tabela."
::= { stereotypeProfileEntry 5 }

statusStereotypeProfile OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"status da entrada. permite por exemplo que a entrada
seja apagada."
::= { stereotypeProfileEntry 6 }

-- Authentication (13)**************************


authenticationNumber OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"Numero de entradas na tabala."
::= { authentication 1 }

authenticationTable OBJECT-TYPE
SYNTAX SEQUENCE OF AuthenticationEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Uma lista com as autenticacoes dos utilizadores."
::= { authentication 2 }

authenticationEntry OBJECT-TYPE
SYNTAX AuthenticationEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Uma entrada que contem informacoes de cada autenticacao."
INDEX { authenticationIndex }
::= { authenticationTable 1 }

AuthenticationEntry ::= SEQUENCE {
authenticationIndex              Unsigned32,
userName                         DisplayString,
password                         DisplayString,
authenticationValue              DisplayString,
```

```
statusAuthentication              Unsigned32
}

authenticationIndex OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada nova
autenticacao.
Os valores atribuidos variam entre 1 e N."
::= { authenticationEntry 1 }

userName OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Username do utilizador."
::= { authenticationEntry 2 }

password OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Password do utilizador."
::= { authenticationEntry 3 }

authenticationValue OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Resultado da autenticacao."
::= { authenticationEntry 4 }

statusAuthentication OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada nova
caracteritica.
Os valores atribuidos devem variar entre 1 e N."
::= { authenticationEntry 5 }


authenticationValueEvent NOTIFICATION-TYPE
```

```
OBJECTS
{
authenticationValue
}
STATUS  current
DESCRIPTION
"parametros a enviar na notificacao."
::= { notificationInterface 1 }


-- complianceGroup (14)************************


grupoSystem OBJECT-GROUP
OBJECTS {
sysDescr,
sysContact,
sysName,
sysLocation     }
STATUS  current
DESCRIPTION
"Grupo de objectos com informacoes sobre o grupo
sistema."
::= { complianceGroup 1 }

grupoUser OBJECT-GROUP
OBJECTS {
userNumber,
userIndex,
userDesignation,
typeUser,
oidLocationUser,
vc,
vl,
va,
statusUser }
STATUS  current
DESCRIPTION
"Grupo de objectos com informacoes sobre o utilizador."
::= { complianceGroup 2 }


grupoUserProfileValues OBJECT-GROUP
OBJECTS {
userProfileValuesNumber,
userProfileValuesIndex,
idxUser,
idxUserProfileComunity,
idxContextUserProfile,
```

```
oidValueUserProfile,
confidenceLevelUserProfile,
statusUserProfile }
STATUS  current
DESCRIPTION
"Grupo de objectos com informacoes sobre os perfis
de utilizador."
::= { complianceGroup 3 }


grupofavoritesInferences OBJECT-GROUP
OBJECTS {
favoritesInferencesNumber,
favoritesInferencesIndex,
idxTypeFavorites,
idxUserFavoritesInferences,
idxContextFavoritesInferences,
oidValueFavoritesInferences,
confidenceLevelFavoritesInferences,
statusFavorites }
STATUS  current
DESCRIPTION
"Grupo de objectos com informacoes sobre as
inferencias de favoritos."
::= { complianceGroup 4 }

grupoInferencesUserActivities OBJECT-GROUP
OBJECTS {
inferencesUserActivitiesNumber,
inferencesUserActivitiesIndex,
idxActivityInferencesUserActivities,
idxUserInferencesUserActivities,
idxContextInferencesUserActivities,
oidValueInferencesUserActivities,
confidenceLevelInferencesUserActivities,
statusUserInferences }
STATUS  current
DESCRIPTION
"Grupo de objectos com informacoes sobre as
inferencias de utilizador."
::= { complianceGroup 5 }



grupostatisticsUserActivities OBJECT-GROUP
OBJECTS {
statisticsUserActivitiesNumber,
statisticsUserActivitiesIndex,
idxActivitystatisticsUserActivities,
```

```
idxUserstatisticsUserActivities,
idxContextstatisticsUserActivities,
oidValuestatisticsUserActivities,
confidenceLevelstatisticsUserActivities,
statusUserstatistics }
STATUS  current
DESCRIPTION
"Grupo de objectos com informacoes sobre as
estatisticas de utilizador."
::= { complianceGroup 6 }


grupoContext OBJECT-GROUP
OBJECTS {
contextNumber,
contextIndex,
idxTemporalContext,
userContext,
oidlocation,
oidEquipmentUsed,
idxInterfaceContext,
statusContext }
STATUS  current
DESCRIPTION
"Grupo de objectos com informacoes sobre o
grupo contexto."
::= { complianceGroup 7 }

grupoTemporalContext OBJECT-GROUP
OBJECTS {
temporalContextNumber,
temporalContextIndex,
beginDateTime,
endDateTime,
monday,
tuesday,
wednesday,
thursday,
friday,
saturday,
sunday,
statusTemporalContext }
STATUS  current
DESCRIPTION
"Grupo de objectos com informacoes sobre o grupo
contexto temporal."
::= { complianceGroup 8 }

grupoIntValues OBJECT-GROUP
```

```
OBJECTS {
intValuesNumber,
intValuesIndex,
valueInt,
statusInt }
STATUS  current
DESCRIPTION
"Grupo de objectos com informacoes sobre o grupo
de valores inteiros."
::= { complianceGroup 9 }

grupoRealValues OBJECT-GROUP
OBJECTS {
realValueNumber,
realValueIndex,
valueReal,
statusReal }
STATUS  current
DESCRIPTION
"Grupo de objectos com informacoes sobre o
grupo de valores reais."
::= { complianceGroup 10 }

grupoStringValues OBJECT-GROUP
OBJECTS {
stringValueNumber,
stringValueIndex,
valueString,
statusString }
STATUS  current
DESCRIPTION
"Grupo de objectos com informacoes sobre o grupo
de valores
do tipo string."
::= { complianceGroup 11 }

grupostereotypeProfile OBJECT-GROUP
OBJECTS {
stereotypeProfileNumber,
stereotypeProfileIndex,
idxStereotype,
idxUserProfileStereotype,
oidValueStereotypeProfile,
confidenceLevelStereotypeProfile,
statusStereotypeProfile }
STATUS  current
DESCRIPTION
"Grupo de objectos com informacoes sobre o grupo
esperiotipo."
```

```
::= { complianceGroup 12 }


grupoAuthentication OBJECT-GROUP
OBJECTS {
authenticationNumber,
authenticationIndex,
userName,
password,
authenticationValue,
statusAuthentication }
STATUS  current
DESCRIPTION
"Grupo de objectos com informacoes sobre o grupo
autenticacao."
::= { complianceGroup 13 }


notificationsGroup NOTIFICATION-GROUP
NOTIFICATIONS {
authenticationValueEvent
 }
STATUS  current
DESCRIPTION
"Grupo de objectos das notificacoes."
::= { notificationInterface 15 }

END
```

MIB interfaces

```
DomoticaSNMPInterfaces-MIB DEFINITIONS ::= BEGIN

IMPORTS

MODULE-IDENTITY, OBJECT-TYPE, NOTIFICATION-TYPE, Counter64,
Unsigned32, Integer32, private, Opaque, TimeTicks,
snmpModules

FROM SNMPv2-SMI
DisplayString, DateAndTime, RowStatus, StorageType,
VariablePointer
FROM SNMPv2-TC
SnmpAdminString
FROM SNMP-FRAMEWORK-MIB
MODULE-COMPLIANCE, OBJECT-GROUP, NOTIFICATION-GROUP
FROM SNMPv2-CONF;


domoticaSNMPMibInterfaceSistema MODULE-IDENTITY
LAST-UPDATED            "201006290000Z"
ORGANIZATION            "DOMinho"
CONTACT-INFO
"Marino Fernandes
email: marinofernandes@hotmail.com"
DESCRIPTION
"MIB para onde serao colocados todos os tipos de interacao
do sistema com a interface"
REVISION                        "201006290000Z"
DESCRIPTION                     "MIB de interfaces."
::= {experimental 80}

--domoticExp    OBJECT IDENTIFIER ::= { private 1 }


--groups in DOMOTICEXP
system
OBJECT IDENTIFIER ::= { domoticaSNMPMibInterfaceSistema 1 }
interfaces
```

```
OBJECT IDENTIFIER ::= { domoticaSNMPMibInterfaceSistema 2 }
featuresInterfaces
OBJECT IDENTIFIER ::= { domoticaSNMPMibInterfaceSistema 3 }
interfaceUser
OBJECT IDENTIFIER ::= { domoticaSNMPMibInterfaceSistema 4 }
menuInterface
OBJECT IDENTIFIER ::= { domoticaSNMPMibInterfaceSistema 5 }
menuFeatures
OBJECT IDENTIFIER ::= { domoticaSNMPMibInterfaceSistema 6 }
typeOfInteractionMenu
OBJECT IDENTIFIER ::= { domoticaSNMPMibInterfaceSistema 7 }
activitiesSubmittedInterface
OBJECT IDENTIFIER ::= { domoticaSNMPMibInterfaceSistema 8 }
activitiesSubmittedInterfaceFeatures
OBJECT IDENTIFIER ::= { domoticaSNMPMibInterfaceSistema 9 }
informationSubmittedInterface
OBJECT IDENTIFIER ::= { domoticaSNMPMibInterfaceSistema 10 }
informationSubmittedInterfaceFeatures
OBJECT IDENTIFIER ::= { domoticaSNMPMibInterfaceSistema 11 }
metadata
OBJECT IDENTIFIER ::= { domoticaSNMPMibInterfaceSistema 12 }
intValues
OBJECT IDENTIFIER ::= { domoticaSNMPMibInterfaceSistema 13 }
realValue
OBJECT IDENTIFIER ::= { domoticaSNMPMibInterfaceSistema 14 }
stringValue
OBJECT IDENTIFIER ::= { domoticaSNMPMibInterfaceSistema 15 }
temporalContext
OBJECT IDENTIFIER ::= { domoticaSNMPMibInterfaceSistema 16 }
complianceGroup
OBJECT IDENTIFIER ::= { domoticaSNMPMibInterfaceSistema 17 }
notificationInterface
OBJECT IDENTIFIER ::= { domoticaSNMPMibInterfaceSistema 18 }


-- SYSTEM (1)************************************

sysDescr OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"Descricao textual da entidade. Inclui informacoes
como nome, versao do software utilizado..."
::= { system 1 }

sysContact OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-only
```

```
STATUS current
DESCRIPTION
"Descricao textual do contacto da pessoa que
desenvolveu o sistema."
::= { system 2 }

sysName OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"Descricao textual com o nome do sistema."
::= { system 3 }

sysLocation OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-only
STATUS current
DESCRIPTION
"Localizacao fisica do sistema. Se a localizacao for
desconhecida, a string fica vazia."
::= { system 4 }


--interfaces (2)***********************************

interfacesNumber OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"Numero de equipamentos existentes neste sistema."
::= { interfaces 1 }

interfacesTable OBJECT-TYPE
SYNTAX SEQUENCE OF InterfacesEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Uma lista com o conjunto de interface que fazem parte
do sistema, correspondendo uma entrada a cada interface."
::= { interfaces 2 }

interfacesEntry OBJECT-TYPE
SYNTAX InterfacesEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Uma entrada que contem informacoes de cada interface."
```

```
INDEX {interfacesIndex}
::= {interfacesTable 1}

InterfacesEntry ::= SEQUENCE {
interfacesIndex                                    Unsigned32,
designationInterface                          DisplayString,
oidLocation                                        DisplayString,
vc                                                              DisplayString,
vl                                                         DisplayString,
va                                                         DisplayString,
ipPortInterfaceNotifications           DisplayString,
statusInterfaces                                   Unsigned32
}

interfacesIndex OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada nova interface.
Os valores atribuidos devem variar entre 1 e N."
::= { interfacesEntry 1 }

designationInterface OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Nome da interface."
::= { interfacesEntry 2 }

oidLocation OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"espaco em que esta inserido a interface."
::= { interfacesEntry 3}

vc OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Comprimento do vector."
::= { interfacesEntry 4 }

vl OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
```

```
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Largura do vector."
::= { interfacesEntry  5}

va OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Altura do vector."
::= { interfacesEntry 6 }

ipPortInterfaceNotifications OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"ip e porta do servidor de notificacoes da interface."
::= { interfacesEntry 7 }


statusInterfaces OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"status da entrada. permite por exemplo que a entrada
seja apagada."
::= { interfacesEntry 8 }

locationInterfaceEvent NOTIFICATION-TYPE
OBJECTS
{
oidLocation
}
STATUS  current
DESCRIPTION
"parametros a enviar na notificacao."
::= { notificationInterface 1 }


--FeaturesInterfaces (3)*****************************


featuresInterfacesNumber OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-write
```

```
STATUS current
DESCRIPTION
"Numero de caracteristicas da tabela."
::= { featuresInterfaces 1 }


featuresInterfacesTable OBJECT-TYPE
SYNTAX SEQUENCE OF FeaturesInterfacesEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Uma lista com o conjunto de caracteristicas."
::= { featuresInterfaces 2 }

featuresInterfacesEntry OBJECT-TYPE
SYNTAX FeaturesInterfacesEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"cada entrada contem informacoes sobre uma caracteristica"
INDEX {featuresInterfacesIndex}
::= { featuresInterfacesTable 1}

FeaturesInterfacesEntry ::= SEQUENCE {
featuresInterfacesIndex                                    Unsigned32,
idxInterface                                                      Unsigned32,
idxCaracteristic                                                  Unsigned32,
oidValueFeatureInterface                                VariablePointer,
statusFeaturesInterfaces                                   Unsigned32
}

featuresInterfacesIndex OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada caracteristicas.
Os valores atribuidos devem variam entre 1 e N."
::= { featuresInterfacesEntry 1 }

idxInterface OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identificador da interface."
::= { featuresInterfacesEntry 2 }

idxCaracteristic OBJECT-TYPE
```

```
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identificador da caracteristica definida pela comunidade
de domotica."
::= { featuresInterfacesEntry 3 }

oidValueFeatureInterface OBJECT-TYPE
SYNTAX VariablePointer
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"apontador para o valor."
::= { featuresInterfacesEntry 4 }

statusFeaturesInterfaces OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"status da entrada. permite por exemplo que a entrada
seja apagada."
::= { featuresInterfacesEntry 5 }

--interfaceUser (4)**********************************

interfaceUserNumber OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"Numero de entradas na tabela equipamento utilizador."
::= { interfaceUser 1 }


interfaceUserTable OBJECT-TYPE
SYNTAX SEQUENCE OF InterfaceUserEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Uma lista com o conjunto de entradas."
::= { interfaceUser 2 }

interfaceUserEntry OBJECT-TYPE
SYNTAX InterfaceUserEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
```

```
"Uma entrada que contem informacoes acerca do utilizador
e interface ligados ao sistema"
INDEX {interfaceUserIndex}
::= { interfaceUserTable 1}

InterfaceUserEntry ::= SEQUENCE {
interfaceUserIndex                                  Unsigned32,
idxUserInterface                                    Unsigned32,
oidUser                                                 DisplayString,
statusInterfaceUser                                 Unsigned32
}


interfaceUserIndex OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada nova entrada.
Os valores atribuidos devem variam entre 1 e N."
::= { interfaceUserEntry 1 }

idxUserInterface OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identificador da interface."
::= { interfaceUserEntry 2 }

oidUser OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identificador do utilizador."
::= { interfaceUserEntry 3 }

statusInterfaceUser OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"status da entrada. permite por exemplo que a entrada
seja apagada."
::= { interfaceUserEntry 4 }


interfaceUserEvent NOTIFICATION-TYPE
```

```
OBJECTS
{
idxUserInterface,
oidUser
}
STATUS  current
DESCRIPTION
"parametros a enviar na notificacao."
::= { notificationInterface 2 }

--Menu (5)***********************************************

menuInterfaceNumber OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"Numero de entradas existentes na tabela."
::= { menuInterface 1 }


menuInterfaceTable OBJECT-TYPE
SYNTAX SEQUENCE OF MenuInterfaceEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Lista com os varias entradas da tabela."
::= { menuInterface 2 }

menuInterfaceEntry OBJECT-TYPE
SYNTAX MenuInterfaceEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Cada entrada contem informacoes sobre os tipos de
inteface aplicaveis ao utilizador."
INDEX {menuInterfaceIndex}
::= {menuInterfaceTable 1}

MenuInterfaceEntry ::= SEQUENCE {
menuInterfaceIndex                                  Unsigned32,
idxInterfaceUsed                                    Unsigned32,
idxInteraction                                      Unsigned32,
statusmenuInterface                                 Unsigned32
}

menuInterfaceIndex OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
```

```
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada novo entrada."
::= { menuInterfaceEntry 1 }

idxInterfaceUsed OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para identificar qual o
equipamento em que a informacao sera apresentada."
::= { menuInterfaceEntry 2 }


idxInteraction OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para identificar qual a
interacao que se pretende apresentar ao utilizador. Ex.
Menu, questao, warning, etc."
::= { menuInterfaceEntry 3 }

statusmenuInterface OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Este campo permite ao agente saber quando deve apagar uma
determinada linha da tabela. Caso o status estiver a 1 a
linha da tabela e apagada."
::= { menuInterfaceEntry 4 }

menuInterfaceEvent NOTIFICATION-TYPE
OBJECTS
{
menuInterfaceIndex,
idxInteraction
}
STATUS  current
DESCRIPTION
"parametros a enviar na notificacao."
::= { notificationInterface 3 }


-- MenuFeatures (6)**********************************
```

```
menuFeaturesNumber OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"Numero de caracteristicas existentes neste sistema."
::= { menuFeatures 1 }

menuFeaturesTable OBJECT-TYPE
SYNTAX SEQUENCE OF MenuFeaturesEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Lista com as varias caracteristicas."
::= { menuFeatures 2 }

menuFeaturesEntry OBJECT-TYPE
SYNTAX MenuFeaturesEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Cada entrada contem informacoes de caracteristicas
aplicaveis a uma unica interface."
INDEX {menuFeaturesIndex}
::= {menuFeaturesTable 1}

MenuFeaturesEntry ::= SEQUENCE {
menuFeaturesIndex                                    Unsigned32,
idxtypeOfInteractionInterface                Unsigned32,
idxFeatureInterface                                  Unsigned32,
oidFeaturesValue                                     VariablePointer,
statusMenuFeatures                                   Unsigned32
}

menuFeaturesIndex OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada nova
caracteritica. Os valores atribuidos devem
variar entre 1 e N ."
::= { menuFeaturesEntry 1 }

idxtypeOfInteractionInterface OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
```

```
"indetifica o menu ao qual pertence a caracteristica."
::= { menuFeaturesEntry 2 }

idxFeatureInterface OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identifica a caracteritica."
::= { menuFeaturesEntry 3 }

oidFeaturesValue OBJECT-TYPE
SYNTAX VariablePointer
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Campo que contem o valor da caracteristica."
::= { menuFeaturesEntry 4 }

statusMenuFeatures OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Este campo permite ao agente saber quando deve
apagar uma determinada linha da tabela. Caso o
status estiver a 1 a linha da tabela e apagada."
::= { menuFeaturesEntry 5 }


--typeOfInteractionMenu (7)**********************


typeOfInteractionMenuNumber OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"Numero de entradas existentes na tabela."
::= { typeOfInteractionMenu 1 }

typeOfInteractionMenuTable OBJECT-TYPE
SYNTAX SEQUENCE OF TypeOfInteractionMenuEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"tabela com os tipo de interacao."
::= { typeOfInteractionMenu 2 }
```

```
typeOfInteractionMenuEntry OBJECT-TYPE
SYNTAX TypeOfInteractionMenuEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Cada entrada contem informacoes de interacao
aplicaveis a ums interacao que esteja a ser
utilizado pelo utilizador."
INDEX {typeOfInteractionMenuIndex}
::= {typeOfInteractionMenuTable 1}

TypeOfInteractionMenuEntry ::= SEQUENCE {
typeOfInteractionMenuIndex                      Unsigned32,
idxMenuTypeOfInteraction                        Unsigned32,
idxTypeOfInteraction                            Unsigned32,
statusTypeOfInteractionMenu                     Unsigned32
}

typeOfInteractionMenuIndex OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada novo
tipo de interacao possivel a apresentar a um
utilizador."
::= { typeOfInteractionMenuEntry 1 }

idxMenuTypeOfInteraction OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada novo
tipo de interacao possivel a apresentar a um
utilizador."
::= { typeOfInteractionMenuEntry 2 }

idxTypeOfInteraction OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada novo
tipo de interacao
possivel a apresentar a um utilizador."
::= { typeOfInteractionMenuEntry 3 }

statusTypeOfInteractionMenu OBJECT-TYPE
```

```
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identificador de utilizador."
::= { typeOfInteractionMenuEntry 4 }


typeOfInteractionMenuEvent NOTIFICATION-TYPE
OBJECTS
{
typeOfInteractionMenuIndex,
idxMenuTypeOfInteraction,
idxTypeOfInteraction
}
STATUS  current
DESCRIPTION
""
::= { notificationInterface 4 }


--      ActivitiesSubmittedInterface (8)********

activitiesSubmittedInterfaceNumber OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"Numero de localizacoes existentes neste sistema."
::= { activitiesSubmittedInterface 1 }

activitiesSubmittedInterfaceTable OBJECT-TYPE
SYNTAX SEQUENCE OF ActivitiesSubmittedInterfaceEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Lista com as varias actividades."
::= { activitiesSubmittedInterface 2 }

activitiesSubmittedInterfaceEntry OBJECT-TYPE
SYNTAX ActivitiesSubmittedInterfaceEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Cada entrada contem informacoes das actividades
que devem ser apresentadas numa determinada interface."
INDEX {activitiesSubmittedInterfaceIndex}
::= {activitiesSubmittedInterfaceTable 1}
```

```
ActivitiesSubmittedInterfaceEntry ::= SEQUENCE {
activitiesSubmittedInterfaceIndex               Unsigned32,
idxMenuInterface                                        Unsigned32,
idxActivity                                                     Unsigned32,
oidEquipament                                           DisplayString,
oidActivitiesSubmittedInterfaceValue    VariablePointer,
preferenceValueAtivity                                  Unsigned32,
oidLocationActivity                                     DisplayString,
actualValueActivity                                     DisplayString,
statusActivitiesSubmittedInterface              Unsigned32
}


activitiesSubmittedInterfaceIndex OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada nova entrada.
Os valores atribuidos devem variar entre 1 e N."
::= { activitiesSubmittedInterfaceEntry 1 }

idxMenuInterface OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada menu onde
devem ser apresentadas as actividades."
::= { activitiesSubmittedInterfaceEntry 2 }

idxActivity OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identifica a actividade."
::= { activitiesSubmittedInterfaceEntry 3 }

oidEquipament OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identifica o equipamento."
::= { activitiesSubmittedInterfaceEntry 4 }

oidActivitiesSubmittedInterfaceValue OBJECT-TYPE
SYNTAX  VariablePointer
```

```
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Campo que contem o valor nominal de uma actividade."
::= { activitiesSubmittedInterfaceEntry 5 }

preferenceValueAtivity OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"preferencia do utilizador pela actividade"
::= { activitiesSubmittedInterfaceEntry 6 }

oidLocationActivity OBJECT-TYPE
SYNTAX DisplayString(SIZE (0..255))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identifica a localizacao da actividade"
::= { activitiesSubmittedInterfaceEntry 7 }

actualValueActivity OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identifica o valor actual da actividade"
::= { activitiesSubmittedInterfaceEntry 8 }

statusActivitiesSubmittedInterface OBJECT-TYPE
SYNTAX Unsigned32 (0..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Este campo permite ao agente saber quando deve apagar
uma determinada linha da tabela. Caso o status estiver
a 2 a linha da tabela e apagada."
::= { activitiesSubmittedInterfaceEntry 9 }

activitiesSubmittedInterfaceEvent NOTIFICATION-TYPE
OBJECTS
{
activitiesSubmittedInterfaceIndex,
idxMenuInterface,
idxActivity,
oidEquipament,
preferenceValueAtivity,
oidLocationActivity,
```

```
actualValueActivity,
statusActivitiesSubmittedInterface
}
STATUS  current
DESCRIPTION
"parametros a enviar na notificacao. Esta notificacao e enviada
quando uma nova actividade e inserida"
::= { notificationInterface 5 }

activitiesSubmittedInterfaceValueEvent NOTIFICATION-TYPE
OBJECTS
{
activitiesSubmittedInterfaceIndex,
oidActivitiesSubmittedInterfaceValue
}
STATUS  current
DESCRIPTION
"parametros a enviar na notificacao. Esta notificacao
e enviada quando um valor e inserido numa actividade"
::= { notificationInterface 6 }


activitiesSubmittedInterfaceNewActualValueEvent
NOTIFICATION-TYPE

OBJECTS
{
activitiesSubmittedInterfaceIndex,
actualValueActivity
}
STATUS  current
DESCRIPTION
"parametros a enviar na notificacao. Esta notificacao e
enviada quando o valor actual de uma actividade e alterado"
::= { notificationInterface 7 }

-- activitiesSubmittedInterfaceFeatures (9)*************

activitiesSubmittedInterfaceFeaturesNumber OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"Numero de caracteristicas atribuidas a uma actividade."
::= { activitiesSubmittedInterfaceFeatures 1 }

activitiesSubmittedInterfaceFeaturesTable OBJECT-TYPE
SYNTAX SEQUENCE OF ActivitiesSubmittedInterfaceFeaturesEntry
MAX-ACCESS not-accessible
```

```
STATUS current
DESCRIPTION
"Lista com as varias caracteristicas."
::= { activitiesSubmittedInterfaceFeatures 2 }

activitiesSubmittedInterfaceFeaturesEntry OBJECT-TYPE
SYNTAX ActivitiesSubmittedInterfaceFeaturesEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Cada entrada contem informacoes de caracteristicas aplicaveis a
uma actividade."
INDEX {activitiesSubmittedInterfaceFeaturesIndex}
::= {activitiesSubmittedInterfaceFeaturesTable 1}

ActivitiesSubmittedInterfaceFeaturesEntry ::= SEQUENCE {
activitiesSubmittedInterfaceFeaturesIndex
Unsigned32,
idxActivitiesSubmittedInterface
Unsigned32,
idxTypeOfInteractionActivitiesInterfaceSubmitted
Unsigned32,
idxFeaturesactivitiesSubmittedInterface
Unsigned32,
oidActivitiesSubmittedInterfaceFeaturesValue
VariablePointer,
statusActivitiesSubmittedInterfaceFeatures
Unsigned32
}

activitiesSubmittedInterfaceFeaturesIndex OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada nova
caracteritica de actividades. Os valores atribuidos
devem variar entre 1 e N."
::= { activitiesSubmittedInterfaceFeaturesEntry 1 }

idxActivitiesSubmittedInterface OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identifica a actividade."
::= { activitiesSubmittedInterfaceFeaturesEntry 2 }

idxTypeOfInteractionActivitiesInterfaceSubmitted
```

```
OBJECT-TYPE

SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identifica o menu e tipo de interacao."
::= { activitiesSubmittedInterfaceFeaturesEntry 3 }

idxFeaturesactivitiesSubmittedInterface OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identifica a caracteristica."
::= { activitiesSubmittedInterfaceFeaturesEntry 4 }

oidActivitiesSubmittedInterfaceFeaturesValue OBJECT-TYPE
SYNTAX VariablePointer
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identifica o valor actual da actividade."
::= { activitiesSubmittedInterfaceFeaturesEntry 5 }

statusActivitiesSubmittedInterfaceFeatures OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Este campo permite ao agente saber quando deve apagar
uma determinada linha da tabela. Caso o status estiver
a 1 a linha da tabela e apagada."
::= { activitiesSubmittedInterfaceFeaturesEntry 6 }

--      informationSubmittedInterface (10)*****************

informationSubmittedInterfaceNumber OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"Numero de entradas na tabela de informacoes para
apresentar ao utilizador existentes."
::= { informationSubmittedInterface 1 }

informationSubmittedInterfaceTable OBJECT-TYPE
SYNTAX SEQUENCE OF InformationSubmittedInterfaceEntry
MAX-ACCESS not-accessible
```

```
STATUS current
DESCRIPTION
"Lista com as varias entradas."
::= { informationSubmittedInterface 2 }

informationSubmittedInterfaceEntry OBJECT-TYPE
SYNTAX InformationSubmittedInterfaceEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"entradas existentes na tabela."
INDEX {informationSubmittedInterfaceIndex}
::= {informationSubmittedInterfaceTable 1}

InformationSubmittedInterfaceEntry ::= SEQUENCE {
informationSubmittedInterfaceIndex              Unsigned32,
idxMenuInterfaceInformation                             Unsigned32,
oidValue                                                        VariablePointer,
description                                                     DisplayString,
statusInformationSubmittedInterface             Unsigned32
}

informationSubmittedInterfaceIndex OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada nova informacao.
Os valores atribuidos devem variar entre 1 e N."
::= { informationSubmittedInterfaceEntry 1 }

idxMenuInterfaceInformation OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identifica o menu."
::= { informationSubmittedInterfaceEntry 2 }

oidValue OBJECT-TYPE
SYNTAX VariablePointer
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identifica o equipamento."
::= { informationSubmittedInterfaceEntry 3 }

description OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
```

```
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"descricao da informacao a apresentar."
::= { informationSubmittedInterfaceEntry 4 }

statusInformationSubmittedInterface OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Este campo permite ao agente saber quando deve apagar
uma determinada linha da tabela. Caso o status estiver
a 1 a linha da tabela e apagada."
::= { informationSubmittedInterfaceEntry 5 }


informationSubmittedInterfaceEvent NOTIFICATION-TYPE
OBJECTS
{
informationSubmittedInterfaceIndex,
idxMenuInterfaceInformation,
description
}
STATUS  current
DESCRIPTION
""
::= { notificationInterface 8 }

-- informationSubmittedInterfaceFeatures (11)**********

informationSubmittedInterfaceFeaturesNumber OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"Numero de caracteristicas existentes neste sistema."
::= { informationSubmittedInterfaceFeatures 1 }

informationSubmittedInterfaceFeaturesTable OBJECT-TYPE
SYNTAX SEQUENCE OF InformationSubmittedInterfaceFeaturesEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Lista com as varias caracteristicas."
::= { informationSubmittedInterfaceFeatures 2 }

informationSubmittedInterfaceFeaturesEntry OBJECT-TYPE
SYNTAX InformationSubmittedInterfaceFeaturesEntry
```

```
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Cada entrada contem informacoes de caracteristicas aplicaveis
a uma informacao a apresentar."
INDEX {informationSubmittedInterfaceFeaturesIndex}
::= {informationSubmittedInterfaceFeaturesTable 1}

InformationSubmittedInterfaceFeaturesEntry ::= SEQUENCE {
informationSubmittedInterfaceFeaturesIndex
Unsigned32,
idxInformationSubmittedInterface
Unsigned32,
typeOfInteractionMenuInformationSubmittedInterface
Unsigned32,
idxFeaturesInformationSubmittedInterface
Unsigned32,
oidinformationSubmittedInterfaceFeaturesValue
VariablePointer,
statusInformationSubmittedInterfaceFeatures
Unsigned32
}

informationSubmittedInterfaceFeaturesIndex OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada nova caracteritica.
Os valores atribuidos devem variar entre 1 e N."
::= { informationSubmittedInterfaceFeaturesEntry 1 }

idxInformationSubmittedInterface OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identifica a informacao."
::= { informationSubmittedInterfaceFeaturesEntry 2 }

typeOfInteractionMenuInformationSubmittedInterface
OBJECT-TYPE

 SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identifica o tipo de interacao."
::= { informationSubmittedInterfaceFeaturesEntry 3 }
```

```
idxFeaturesInformationSubmittedInterface OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"indeitifica a caracteristica."
::= { informationSubmittedInterfaceFeaturesEntry 4 }

oidinformationSubmittedInterfaceFeaturesValue OBJECT-TYPE
SYNTAX VariablePointer --DisplayString (SIZE (0..255))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Campo que contem o valor nominal de uma caracteristica."
::= { informationSubmittedInterfaceFeaturesEntry 5 }

statusInformationSubmittedInterfaceFeatures OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Este campo permite ao agente saber quando deve apagar
uma determinada linha da tabela. Caso o status estiver
a 1 a linha da tabela e apagada."
::= { informationSubmittedInterfaceFeaturesEntry 6 }


-- metadata (12)**************************************

metadataNumber OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"Numero de informacoes na tabela."
::= { metadata 1 }

metadataTable OBJECT-TYPE
SYNTAX SEQUENCE OF MetadataEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Lista com as varias entradas da tabela."
::= { metadata 2 }

metadataEntry OBJECT-TYPE
SYNTAX MetadataEntry
MAX-ACCESS not-accessible
```

```
STATUS current
DESCRIPTION
"Cada entrada contem informacoes de cada meta-informacao."
INDEX {metadataIndex}
::= {metadataTable 1}

MetadataEntry ::= SEQUENCE {
metadataIndex                                   Unsigned32,
idxMenuMetadata                                 Unsigned32,
idxActivityMetadata                             Unsigned32,
idxFavoritType                                  Unsigned32,
metadataValues                                  DisplayString,
metadataRestrictions                        DisplayString,
statusMetadata                                  Unsigned32
}


metadataIndex OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada meta-informacao.
Os valores atribuidos devem variar entre 1 e N."
::= { metadataEntry 1 }

idxMenuMetadata OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identifica o menu ao qual pertence a meta-informacao."
::= { metadataEntry 2 }

idxActivityMetadata OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identifica a actividade ao qual pertence a meta-informacao."
::= { metadataEntry 3 }

idxFavoritType OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identifica o favorito definido pela comunidade de domotica
ao qual pertence a meta-informacao."
```

```
::= { metadataEntry 4 }

metadataValues OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identifica os valores da meta-informacao."
::= { metadataEntry 5 }

metadataRestrictions OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"identifica os valores restritos a meta-informacao."
::= { metadataEntry 6 }

statusMetadata OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Este campo permite ao agente saber quando deve apagar uma
determinada linha da tabela. Caso o status estiver a 1 a linha
da tabela e apagada."
::= { metadataEntry 7 }


metadataEvent NOTIFICATION-TYPE
OBJECTS
{
metadataIndex,
idxMenuMetadata,
idxActivityMetadata,
idxFavoritType,
metadataValues,
metadataRestrictions
}
STATUS  current
DESCRIPTION
""
::= { notificationInterface 9 }

-- intValues (13)************************************


intValuesNumber OBJECT-TYPE
SYNTAX Unsigned32
```

```
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"Numero de valores inteiros diferentes existentes neste
sistema."
::= {intValues 1 }


intValuesTable OBJECT-TYPE
SYNTAX SEQUENCE OF IntValuesEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Uma lista com o conjunto de valores inteiros."
::= { intValues 2 }

intValuesEntry OBJECT-TYPE
SYNTAX IntValuesEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Uma entrada que contem informacoes de gestao"
INDEX {intValuesIndex}
::= { intValuesTable 1}

IntValuesEntry ::= SEQUENCE {
intValuesIndex                          Unsigned32,
valueInt                        Integer32,
statusInt                                               Unsigned32
}

intValuesIndex OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada novo valor. Os
valores atribuidos devem variam entre 1 e N."
::= { intValuesEntry 1 }

valueInt OBJECT-TYPE
SYNTAX Integer32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"descricao dos tipos de valores possiveis."
::= { intValuesEntry 2 }

statusInt OBJECT-TYPE
```

```
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"status da entrada. permite por exemplo que a entrada
seja apagada."
::= { intValuesEntry 3 }

-- realValue (14)************************************

realValueNumber OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"Numero de valores reais diferentes existentes neste
sistema."
::= {realValue 1 }


realValueTable OBJECT-TYPE
SYNTAX SEQUENCE OF RealValueEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Uma lista com o conjunto de valores reais."
::= { realValue 2 }

realValueEntry OBJECT-TYPE
SYNTAX RealValueEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Uma entrada que contem informacoes de gestao"
INDEX {realValueIndex}
::= { realValueTable 1}

RealValueEntry ::= SEQUENCE {
realValueIndex                  Unsigned32,
valueReal                       DisplayString,
statusReal                                              Unsigned32
}

realValueIndex OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada novo valor.
```

```
Os valores atribuidos devem variam entre 1 e N."
::= { realValueEntry 1 }

valueReal OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"descricao dos tipos de valores possiveis."
::= { realValueEntry 2 }

statusReal OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"status da entrada. permite por exemplo que a entrada
seja apagada."
::= { realValueEntry 3 }
-- stringValue (15)*********************************

stringValueNumber OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"Numero de valores do tipo string diferentes existentes
neste sistema."
::= {stringValue 1 }


stringValueTable OBJECT-TYPE
SYNTAX SEQUENCE OF StringValueEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Uma lista com o conjunto de valores do tipo string."
::= { stringValue 2 }

stringValueEntry OBJECT-TYPE
SYNTAX StringValueEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Uma entrada que contem informacoes de gestao"
INDEX {stringValueIndex}
::= { stringValueTable 1}

StringValueEntry ::= SEQUENCE {
```

```
stringValueIndex                Unsigned32,
valueString                     DisplayString,
statusString                                    Unsigned32
}

stringValueIndex OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada novo valor.
Os valores atribuidos devem variam entre 1 e N."
::= { stringValueEntry 1 }

valueString OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"descricao dos tipos de valores possiveis."
::= { stringValueEntry 2 }

statusString OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"status da entrada. permite por exemplo que a entrada seja
apagada."
::= { stringValueEntry 3 }

-- temporalContext (16)**********************************

temporalContextNumber OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-write
STATUS current
DESCRIPTION
"Numero de contextos temporais existentes na tabela."
::= { temporalContext 1 }

temporalContextTable OBJECT-TYPE
SYNTAX SEQUENCE OF TemporalContextEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Lista com os varios contextos temporais."
::= { temporalContext 2 }
```

```
temporalContextEntry OBJECT-TYPE
SYNTAX TemporalContextEntry
MAX-ACCESS not-accessible
STATUS current
DESCRIPTION
"Cada entrada contem informacoes de cada contexto temporal."
INDEX {temporalContextIndex}
::= {temporalContextTable 1}

TemporalContextEntry ::= SEQUENCE {
temporalContextIndex             Unsigned32,
beginDateTime                    DisplayString,
endDateTime                      DisplayString,
monday                           Unsigned32,
tuesday                          Unsigned32,
wednesday                        Unsigned32,
thursday                                 Unsigned32,
friday                           Unsigned32,
saturday                         Unsigned32,
sunday                           Unsigned32,
statusTemporalContext                Unsigned32
}

temporalContextIndex OBJECT-TYPE
SYNTAX Unsigned32
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada novo
contexto temporal. Os valores atribuidos devem
variar entre 1 e N."
::= { temporalContextEntry 1 }

beginDateTime OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"data e hora de inicio do contexto."
::= { temporalContextEntry 2 }


endDateTime OBJECT-TYPE
SYNTAX DisplayString (SIZE (0..255))
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"data e hora de fim do contexto."
::= { temporalContextEntry 3 }
```

```
monday OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"todas as segundas-feiras."
::= { temporalContextEntry 4 }

tuesday OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"todas as tercas-feiras."
::= { temporalContextEntry 5 }

wednesday OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"todas as quartas-feiras."
::= { temporalContextEntry 6 }


thursday OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"todas as quintas-feiras."
::= { temporalContextEntry 7 }

friday OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"todas as sextas-feiras."
::= { temporalContextEntry 8 }

saturday OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"todas os sabados."
```

```
::= { temporalContextEntry 9 }

sunday OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"todas as domingos."
::= { temporalContextEntry 10 }

statusTemporalContext OBJECT-TYPE
SYNTAX Unsigned32 (1..2)
MAX-ACCESS read-create
STATUS current
DESCRIPTION
"Um unico valor maior que zero para cada nova
caracteritica. Os valores atribuidos devem variar
entre 1 e N caracteriticas."
::= { temporalContextEntry 11 }


-- complianceGroup (17)************************


grupoSystem OBJECT-GROUP
OBJECTS {
sysDescr,
sysContact,
sysName,
sysLocation     }
STATUS  current
DESCRIPTION
"Grupo de objectos com informacoes sobre o grupo
sistema."
::= { complianceGroup 1 }

grupoInterfaces OBJECT-GROUP
OBJECTS {
interfacesNumber,
interfacesIndex,
designationInterface,
oidLocation,
vc,
vl,
va,
ipPortInterfaceNotifications,
statusInterfaces        }
STATUS  current
DESCRIPTION
```

```
"Grupo de objectos com informacoes sobre o grupo
interfaces."
::= { complianceGroup 2 }

grupoFeaturesInterfaces OBJECT-GROUP
OBJECTS {
featuresInterfacesNumber,
featuresInterfacesIndex,
idxInterface,
oidLocation,
idxCaracteristic,
oidValueFeatureInterface,
statusFeaturesInterfaces }
STATUS  current
DESCRIPTION
"Grupo de objectos com informacoes sobre o grupo
caracteristicas de interfaces."
::= { complianceGroup 3 }

grupoInterfaceUser OBJECT-GROUP
OBJECTS {
interfaceUserNumber,
interfaceUserIndex,
idxUserInterface,
oidUser,
statusInterfaceUser     }
STATUS  current
DESCRIPTION
"Grupo de objectos com informacoes sobre o grupo
interface utilizador."
::= { complianceGroup 4 }


grupoMenuInterface OBJECT-GROUP
OBJECTS {
menuInterfaceNumber,
menuInterfaceIndex,
idxInterfaceUsed,
idxInteraction,
statusmenuInterface     }
STATUS  current
DESCRIPTION
"Grupo de objectos com informacoes sobre o grupo
menu interface."
::= { complianceGroup 5 }

grupoMenuFeatures OBJECT-GROUP
OBJECTS {
menuFeaturesNumber,
```

```
menuFeaturesIndex,
idxtypeOfInteractionInterface,
idxFeatureInterface,
oidFeaturesValue,
statusMenuFeatures      }
STATUS  current
DESCRIPTION
"Grupo de objectos com informacoes sobre o grupo
caracteristicas de menu."
::= { complianceGroup 6 }


grupoTypeOfInteractionMenu OBJECT-GROUP
OBJECTS {
typeOfInteractionMenuNumber,
typeOfInteractionMenuIndex,
idxMenuTypeOfInteraction,
idxTypeOfInteraction,
statusTypeOfInteractionMenu }
STATUS  current
DESCRIPTION
"Grupo de objectos com informacoes sobre o grupo
menu tipo de interacao."
::= { complianceGroup 7 }


grupoActivitiesSubmittedInterface OBJECT-GROUP
OBJECTS {
activitiesSubmittedInterfaceNumber,
activitiesSubmittedInterfaceIndex,
idxMenuInterface,
idxActivity,
oidEquipament,
oidActivitiesSubmittedInterfaceValue,
preferenceValueAtivity,
oidLocationActivity,
actualValueActivity,
statusActivitiesSubmittedInterface      }
STATUS  current
DESCRIPTION
"Grupo de objectos com informacoes sobre o grupo
actividades a apresentar ao utilizador."
::= { complianceGroup 8 }

grupoActivitiesSubmittedInterfaceFeatures OBJECT-GROUP
OBJECTS {
activitiesSubmittedInterfaceFeaturesNumber,
activitiesSubmittedInterfaceFeaturesIndex,
idxActivitiesSubmittedInterface,
```

```
idxTypeOfInteractionActivitiesInterfaceSubmitted,
idxFeaturesactivitiesSubmittedInterface,
oidActivitiesSubmittedInterfaceFeaturesValue,
statusActivitiesSubmittedInterfaceFeatures      }
STATUS  current
DESCRIPTION
"Grupo de objectos com informacoes sobre o grupo
caracteristicas de actividades a apresentar ao
utilizador."
::= { complianceGroup 9 }


grupoInformationSubmittedInterface OBJECT-GROUP
OBJECTS {
informationSubmittedInterfaceNumber,
informationSubmittedInterfaceIndex,
idxMenuInterfaceInformation,
oidValue,
description,
statusInformationSubmittedInterface     }
STATUS  current
DESCRIPTION
"Grupo de objectos com informacoes sobre o grupo
informacoes a apresentar ao utilizador."
::= { complianceGroup 10 }


grupoInformationSubmittedInterfaceFeatures OBJECT-GROUP
OBJECTS {
informationSubmittedInterfaceFeaturesNumber,
informationSubmittedInterfaceFeaturesIndex,
idxInformationSubmittedInterface,
typeOfInteractionMenuInformationSubmittedInterface,
idxFeaturesInformationSubmittedInterface,
oidinformationSubmittedInterfaceFeaturesValue,
statusInformationSubmittedInterfaceFeatures     }
STATUS  current
DESCRIPTION
"Grupo de objectos com informacoes sobre o grupo
caracteristicas de informacoes a apresentar ao utilizador."
::= { complianceGroup 11 }


grupoMetadata OBJECT-GROUP
OBJECTS {
metadataNumber,
metadataIndex,
idxMenuMetadata,
idxActivityMetadata,
```

```
idxFavoritType,
metadataValues,
metadataRestrictions,
statusMetadata }
STATUS  current
DESCRIPTION
"Grupo de objectos com informacoes de meta-informacao."
::= { complianceGroup 12 }

grupoIntValues OBJECT-GROUP
OBJECTS {
intValuesNumber,
intValuesIndex,
valueInt,
statusInt       }
STATUS  current
DESCRIPTION
"Grupo de objectos com informacoes sobre o grupo de
valores inteiros."
::= { complianceGroup 13 }

grupoRealValues OBJECT-GROUP
OBJECTS {
realValueNumber,
realValueIndex,
valueReal,
statusReal      }
STATUS  current
DESCRIPTION
"Grupo de objectos com informacoes sobre o grupo de
valores reais."
::= { complianceGroup 14 }

grupoStringValues OBJECT-GROUP
OBJECTS {
stringValueNumber,
stringValueIndex,
valueString,
statusString    }
STATUS  current
DESCRIPTION
"Grupo de objectos com informacoes sobre o grupo de
valores do tipo string."
::= { complianceGroup 15 }


grupoTemporalContext OBJECT-GROUP
OBJECTS {
temporalContextNumber,
```

```
temporalContextIndex,
beginDateTime,
endDateTime,
monday,
tuesday,
wednesday,
thursday,
friday,
saturday,
sunday,
statusTemporalContext }
STATUS  current
DESCRIPTION
"Grupo de objectos com informacoes sobre o grupo
contexto temporal."
::= { complianceGroup 16 }

-- notificationsGroup (18)**********************

notificationsGroup NOTIFICATION-GROUP
NOTIFICATIONS {
locationInterfaceEvent,
interfaceUserEvent,
menuInterfaceEvent,
typeOfInteractionMenuEvent,
activitiesSubmittedInterfaceEvent,
activitiesSubmittedInterfaceValueEvent,
activitiesSubmittedInterfaceNewActualValueEvent,
informationSubmittedInterfaceEvent,
metadataEvent }
STATUS  current
DESCRIPTION
"Grupo de objectos das notificacoes."
::= { notificationInterface 10 }

END
```